Neural Architectures for Named Entity Recognition and Relation Classification in Biomedical and Clinical Texts

Sunil Kumar Sahu

Neural Architectures for Named Entity Recognition and Relation Classification in Biomedical and Clinical Texts

Thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

by

Sunil Kumar Sahu

Under the supervision of

Dr. Ashish Anand

Department of Computer Science and Engineering

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

Guwahati 781039, India

June 08, 2018

Declaration

I certify that

a. The work contained in this thesis is original, and has been done by myself under the general supervision of my supervisors.

b. The work has not been submitted to any other institute for any degree or diploma.

c. Whenever I have used materials (data, theoretical analysis, results) from other sources, I have given due credit to them by citing them in the text of the thesis and giving their details in the references.

d. Whenever I have quoted written materials from other sources, I have put them under quotation marks and given due credit to the sources by citing them and giving required details in the references.

Place: IIT Guwahati
Date:

Sunil Kumar Sahu
Research Scholar
Department of Computer Science
and Engineering,
Indian Institute of Technology Guwahati,
Guwahati 781039, India

CERTIFICATE

This is to certify that this thesis entitled "**Neural Architectures for Named Entity Recognition and Relation Classification in Biomedical and Clinical Texts**" being submitted by Mr. Sunil Kumar Sahu to the Department of Computer Science and Engineering, Indian Institute of Technology Guwahati, is a record of bona fide research work under my supervision and is worthy of consideration for the award of the degree of Doctor of Philosophy of the Institute.

The results contained in this thesis have not been submitted in part or full to any other university or institute for the award of any degree or diploma.

Dr. Ashish Anand
Dept. of Computer Sc. and Engg.
Indian Institute of Technology Guwahati
Guwahati-781039, Assam, India
Place: IIT Guwahati
Date:

Acknowledgements

First, I would like to convey my deepest gratitude to my parents, whose love, support, encouragement, inspiration, and sacrifice have made my success possible. Their simple but respectable lifestyle has provided me the liberty to be fearless and explore new things in life. Probably, I would have never decided to pursue a PhD, and able to submit my thesis, if they were otherwise. I also thank my other family members, especially my elder brother *Mr. Santosh Kumar Sahu*, for their unconditional love and support, which they have been sharing with me throughout my life.

I want to thank my advisor, *Dr. Ashish Anand*. He taught me how to do research. He is an excellent mentor who gave me the freedom and the encouragement to pursue my ideas and the opportunities to attend conferences and workshops. More importantly, he gave me his valuable help and support whenever it was needed. I am grateful for being his student. I am also profoundly grateful to my Doctoral Committee members, *Prof. S.R.M. Prasanna*, *Dr. V. Vijaya Saradhi*, and *Dr. Sanasam Ranbir Singh* for sharing their insightful comments and suggestions on my research work. I would also like to thank all my collaborators *Muneeb TH*, *Vishal Anand*, *Ranti Dev Sharma*, *Salama*, *Rijil*, *Rahul Patchigolla*, *Desh Raj* and *Kushal Chawala* for collaborating with me during my PhD tenure. I also want to acknowledge my colleagues *Paul* and *Austin* from the NaCTeM lab and *Saptarshi*, *Abhishek*, *Neelakshi* and *Akshay* from IIT Guwahati for proofreading the thesis.

I would like to thank the present Head of the Department of Computer Science and Engineering, for providing an excellent research environment

in the department, and support my research works in many ways. I also thank the department's scientific staffs *Mr. Bhriguraj Bora* and *Mr. Nanu A. Kachari* for extending their helping hands in solving many technical issues which I faced during my PhD.

I am thankful for organizations such as Microsoft Research India and Google India for sponsoring my travel to attend international conferences. In addition, I would like to thank MHRD, Govt. of India, for supporting my research through a PhD fellowship.

I would also like to acknowledge I2B2-2010 and SemEval-2013 for providing the dataset originally prepared for the shared tasks, clinical information extraction and drug-drug interaction extraction respectively. Moreover, I would like to acknowledge the use of computing resources made available from the Board of Research in Nuclear Science (BRNS), Dept of Atomic Energy (DAE) Govt. of India sponsored project (No.2013/13/8-BRNS/10026) by *Dr. Aryabartta Sahu* at Department of Computer Science and Engineering, IIT Guwahati.

At last but not least, I express my sincere thanks to *Dr. Sandya Mannarswamy*, Research Scientist at Xerox Research Center India, Bangalore and *Dr. Krishnadev Oruganty*, Research Scientist at GVKBioscience, Hyderabad for guiding me during my internships in respective places.

Place: IIT Guwahati

Date: **Sunil Kumar Sahu**

Abstract

The increasing number of biomedical and clinical texts such as research articles, discharge summaries, electronic health records and texts created by social network users is an immeasurable source of information. The extracted information can be used for several applications, *e.g.*, construction of medical knowledge bases, drug repurposing etc. Extracting structured information from unstructured text is called *information extraction* (IE) and is considered as a higher level of natural language processing (NLP) task. Regular organization of shared challenges for the last decade for various information extraction tasks in the biomedical domain has made several standard benchmark datasets publicly available. Availability of the benchmark datasets has led to a continuous development of various methods for information extraction tasks. The majority of existing methods divide IE tasks into several subtasks. *Named entity recognition* (NER), and *relation classification* (RC) are the two main subtasks. In each subtask, explicitly designed features are used in machine learning (ML) methods for classification into correct categories. Although ML methods have been successfully used for many biomedical NER and RC tasks, they still face a few challenges. The performance of such methods is highly dependent on the quality of user-designed features. Further, these feature sets also need to be adapted if domain or task is changed from one to another. For instance, a set of morphological feature designed for gene entity recognition may not work for drug or disease name recognition and features designed based on lexical resources for gene entity recognition may not be suitable for disease name

recognition. Other features may require domain-specific resources or NLP tools. Another major challenge faced is in making the whole system reproducible and usable in practice. This happens due to the lack of finer details of feature engineering available in the public domain.

Recent years have seen renewed interest in *representation learning* using neural network models. One of the primary motivations of such models is to reduce the efforts required for explicit feature engineering. *Representation learning* is a way to learn the projection of the data that helps a machine learning model to make the correct prediction. For instance, in an NER task, a good projection is one which embeds linguistics, orthographic, contextual and syntactic information of a word with its representation. Similarly, in an RC task, a good projection would be one which embeds semantic and syntactic information about the sentence with targeted entities. In this thesis, we focus on these two subtasks of IE. Our objective is to use representation learning with reduced explicit feature engineering to benchmark against standard approaches and to analyze the results. Towards this end, we employ several neural network models and analyze their performances on the two subtasks of IE.

Firstly, we focus on diverse entity types occurring in two different text sources, where the nature of the text also differs. In particular, we classify disease and drug entities present in abstracts of biomedical research articles, and clinical entities appearing in discharge summaries or clinical notes. In both scenarios, our objective is to use the same set of features without utilizing any task or domain-specific resources. Towards this objective, we propose a model based on a bi-directional long short-term memory network (BLSTM) for different biomedical entity recognition tasks. Our model uses two different BLSTMs. The first BLSTM works on characters of each word in a given sequence to learn morphologically rich feature vectors, whereas the second BLSTM works at the word level. Both BLSTMs together try to

learn contextually rich feature vectors for each word in the sequence. The extracted feature vectors are used to predict entities in the sequence using a conditional random field (CRF) layer. Our results indicate that the same model can achieve state-of-the-art results in this manner even for diverse entity types appearing in a different genre of texts. Motivated by the high level of performance in the NER task, we subsequently explore convolution neural networks (CNN) and BLSTM networks in multiple biomedical RC tasks. Here models use raw text (only word and sentence segmentation has been done as pre-processing) with targeted entities as their input and they predict either a correct class of relation or no relation as their output. Extensive analysis performed on drug-drug interaction (DDI) extraction and clinical relation classification (CRC) tasks show the following: state-of-the-art results can be achieved, and LSTM models are likely to perform better than CNN models, especially for identifying relations in longer sentences. Finally, we explore whether a model trained on one task can be utilized for another task. Our main motivation comes from the practical issue of generating a sufficient amount of training and test data for a particular task. We propose three methods for utilizing the knowledge learned from a *source task*, where we have sufficient training data, to a *target task*, where we do not have sufficient training data. We systematically investigate the effectiveness of the proposed methods in transferring the knowledge in multiple ways related to different biomedical RC tasks, such as similarity or relatedness between the source and target tasks, and the size of training data for the source task.

Across the two subtasks of NER and RC, all proposed neural network models are systematically analyzed. The analysis is undertaken keeping in mind multiple aspects, such as the usefulness of representation learning, the advantages of adding additional features, *e.g.* POS tags, and error analysis of the models. In all models, features are appropriately represented by a vector which is learned during training. These vector representations, called *latent features*, work as learned discriminative

features. Further, through our experiments, we also show that initializing the vector representation of each word with pre-trained vectors improves the performance of the models for both the tasks. Pre-trained word vectors are also obtained from an in-house pre-processed PubMed corpus using different word embedding techniques. All the proposed models are generic, end-to-end (almost raw text to prediction) and use latent features in place of manually defined features. We observe in many tasks that, such models achieve new state-of-the-art performance or otherwise achieve a performance that is competitive with respect to the current state-of-the-art.

Contents

List of Tables

LIST OF TABLES

List of Figures

List of Symbols

NLP	Natural Language Processing
NER	Named Entity Recognition
BNER	Biomedical Named Entity Recognition
Clinical NER	Clinical Named Entity Recognition
Drug NER	Drug Named Entity Recognition
Disease NER	Disease Named Entity Recognition
BioNLP	Biomedical Natural Language Processing
RE	Relation Extraction
DDI	Drug Drug Interaction
DDIC	Drug Drug Interaction Classification
CRE	Clinical Relation Extraction
TL	Transfer Learning
IE	Information Extraction
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory Network
BLSTM	Bidirectional Long Short Term Memory Network
GRU	Gated Recurrent Unit
CNN	Convolution Neural Network
DNN	Deep Neural Network
SVM	Support Vector Machine

LIST OF SYMBOLS

CRF	Conditional Random Field
HMM	Hidden Markov Model
MEMM	Maximum Entropy Markov Model
UMLS	Unified Medical Language System
EHR	Electronic Health Record
I2B2	Informatics for Integrating Biology and the Bedside
FN	False Negative
FP	False Positive
ADR	Adverse Drug Reaction
NLM	US National Library of Medicine
PoS	Part-Of-Speech
IIT	Indian Institute of Technology

Chapter 1

Introduction

Biomedical and clinical texts such as research articles, clinical trials, discharge summaries, and other texts created by social network users, represent a large source of information. For example, drug-drug interaction (DDI) information, adverse drug reaction information, drug-disease association information, etc. can be obtained from them. The extracted information can then be used for the curation of a medical knowledge base, drug repurposing and other information retrieval tasks [Ananiadou et al., 2006, Shang et al., 2011, Uzuner et al., 2014]. Although several relevant manually curated datasets exist, keeping them updated with the increasing rate of biomedical literature is a challenging task [Segura Bedmar et al., 2011, Segura-Bedmar et al., 2013]. PubMed, a database of biomedical articles, contains about 27 million citations[1] and approximately 0.8 million articles are added annually[2]. Manually transforming this information from human-readable text to machine-readable format is time-consuming and laborious. Therefore, it is important to make an intelligent or automatic system which can read, understand and extract relevant information from the unstructured or noisy text and create a machine-readable knowledge base. Automatically extracting entities and their relationships

[1] `https://www.ncbi.nlm.nih.gov/pubmed`

[2] `https://www.nlm.nih.gov/bsd/stats/cit_added.html`

from the unstructured or semi-structured text is a part of IE and considered as a higher level NLP task.

Conventional NLP techniques build a model for these tasks based on statistical or manually-designed features extracted from the training dataset. These features are designed based on observed patterns obtained by analyzing the training dataset. Most of these features are obtained from the output of other existing NLP tools, and some of them require domain-specific lexical recourses. This kind of explicit feature engineering is task and domain-specific, and requires a large amount of trial and error analysis to obtain a high-quality feature set having strong discriminative characteristics. Moreover, commonly used models such as perceptron [Rosenblatt, 1958], linear SVM [Cortes and Vapnik, 1995] and linear CRF [Lafferty et al., 2001], are shallow or linear in nature. The performance of the shallow models are heavily dependent on the choice of good quality feature sets and are adversely affected if poorly designed features are used.

On the other hand, *representation* or *feature learning* is a way to obtain a suitable representation of data, which helps a model to make the correct prediction. For instance, in the NER task, a good representation of a word must embed linguistics, orthographic, contextual and syntactic information. All these types of information are important clues for identifying entities in the text. *Representation learning* helps in avoiding the need for manual feature engineering. The current advancement in the optimization and regularization of neural network models [Hinton et al., 2006, Kingma and Ba, 2014, Srivastava et al., 2014] as well as technological progress in high-performance computing has resulted in the neural network becoming a robust model for feature learning. Neural networks have shown a powerful feature learning ability and have achieved appreciable results in the different *computer vision* [Krizhevsky et al., 2012, Karpathy and Fei-Fei, 2014], *speech recognition* [Dahl et al., 2012, Hinton et al., 2012] and a variety of NLP

[Collobert et al., 2011, Kim, 2014] tasks.

This thesis takes its motivation from the above discussion and explores neural network models for two classical problems of biomedical IE: *biomedical entity recognition* and *relation classification*. All the proposed models use representation learning in place of explicit feature engineering and learn a suitable representation while training the models. To begin with, we propose a model based on different neural network architectures for the task of biomedical entity recognition. The proposed model is evaluated on drug name recognition (drug NER), disease name recognition (disease NER) and clinical entity recognition (clinical NER) tasks. The corpus for the disease NER contains abstracts of biomedical research articles, whereas a collection of discharge summaries is used as a corpus for the clinical NER. Although the corpora and nature of the text within them are task specific and different from each other, the proposed models use the same set of features with no domain-specific knowledge. The significant improvement over baseline methods shows the importance of representation learning. Next, we propose different models for relation classification tasks. Each of the proposed models takes a sentence and targeted entities within the sentence as inputs and predicts either one of the relation classes or no relation as their output. All of the proposed models are evaluated on two different relation classification tasks, namely, clinical relation classification (CRC) and drug relation classification (DDI Extraction). Here again, a collection of discharge summaries is used as the corpus for the *CRC* task and abstracts of biomedical research articles and documents from DrugBank are used for the DDI Extraction task. The experimental results show that the proposed model outperforms existing feature-based methods.

Although the CRC and DDI extraction tasks are analogous, traditional methods require a sufficient number of annotated training instances for each of these tasks. We have already observed that even without any domain or task-

specific features, high level of performance can be achieved using neural network models. Therefore, in the last part of the thesis, we explore whether knowledge gained in one task can be utilized in other task. If the results are competitive, it will reduce the time and effort required to prepare annotated training datasets. To this end, we propose three frameworks for transferring the knowledge gained from one task into another (Transfer Learning (TL)) in different relation classification tasks. We show the efficacy of TL frameworks in various biomedical and clinical relation classification tasks. Our results demonstrate that in general, a TL model helps in improving the performance, although the extent of improvement depends on the size of the dataset and the relatedness of two tasks.

We firstly describe each task and its formulation in Sec. 1.1. Sec. 1.2 briefly discusses existing methods used for each of the tasks. The contribution of the thesis is highlighted in Sec. 1.3.

1.1 Problem Formulation

In this section, we introduce a few necessary concepts that are required to describe NER and RE tasks. Subsequently, we discuss biomedical NER and RE, and their formulations.

Nomenclature

- **Classification Problem**: In machine learning, a classification problem is the task of assigning a class from a predefined fixed set C of classes to an input pattern. This is one of the core problems in machine learning and has a large variety of practical applications in NLP.

- **Classification Problem as a Supervised Learning Problem** : Given a training set, $D = \{(x_1, y_1), \cdots ,(x_n, y_n)\}$, where $x_i \in \mathbb{R}^d$ is a given input

pattern, $y_i \in Y$ is its class and n indicates a number of training instances. A supervised learning model will learn a hypothesis $g : X \rightarrow Y$ from the dataset D for mapping an input $x \in X$ to its class $y \in Y$. Here, X and Y represent the input and output spaces respectively.

- **Sequence Labeling Problem** : Sequence labeling is the task of assigning a label to an element of a sequence from a predefined label set L. More formally, given a sequence of input tokens $s = x_1 \cdots x_n$ and a set of labels $L = \{l_1, \cdots l_k\}$, the sequence labeling problem is to determine the correct sequence of labels $a = t_1 \cdots t_n$ such that $t_i \in L$ for $1 \leq i \leq n$. Many problems in NLP can be formulated as sequence labeling problems, *e.g.*, parts of speech (PoS) tagging, chunking, NER etc.

- **Sequence Labeling as a Supervised Learning Problem** : Given a training set $D = \{(s_1, a_1) \cdots (s_n, a_n)\}$, where $s_i = x_1 \cdots x_{n_i}$ is a word sequence, $a_i = t_1 \cdots t_{n_i}$ is its label sequence and n_i is the length of i^{th} sequence. A supervised learning model will learn a hypothesis $g : S \rightarrow A$, for mapping an input $s \in S$, representing a word sequence to an output $a \in A$, representing its label sequence. Here, S and A represent the input and output spaces respectively.

- **BIO Tagging Scheme**: BIO tagging is the simplest and most widely used tagging scheme employed for the sequence labeling task. It is applicable to multiclass and multi-word entity recognition tasks. It maintains two labels, *B-Entity* and *I-Entity* for each class of entity and an *O* (outside) label in its label set [Settles, 2004]. In this scheme a token would be tagged with the label *B-Entity* if it is the first word of an entity phrase, and it would be assigned *I-Entity* if it is a token between the second and the last word of the entity phrase. Otherwise, it would be tagged with the *O* label.

Named Entity Recognition

Named entity recognition addresses the problem of identifying predefined classes of named entities in text. In the biomedical domain, an entity can be a drug, disease, treatment, test, etc. In literature, the named entity recognition task is modeled as a *sequence labeling* problem, where every word in a text is assigned a label from a predefined set. All the entities and their classes are identified based on assigned labels. Table 1.1 shows an example of a clinical entity recognition task, where *lasix* and *congestive heart failure* are entities of type *Treatment* and *Problem* respectively.

He	*was*	*given*	*lasix*	*to*	*prevent*	*him*	*from*	*congestive*	*heart*	*failure*	.
O	O	O	B-Treat	O	O	O	O	B-Prob	I-Prob	I-Prob	O

Table 1.1: Example of a clinical entity recognition task using the BIO tagging scheme.

Relation Classification

Relation classification (RC) is the task of identifying how given entities are semantically related in a text. In the biomedical domain, the relation can be an interaction between drugs, the association of a problem and a treatment, etc. The example shown in Table 1.1 shows a relation of type *treatment administered for medical problem* between the two entities *lasix* and *congestive heart failure*. For the relation classification task, we assume that entities have been already identified and the goal is to determine the type of semantic relation between the given entities. In general, RC tasks are more difficult than sequence labeling tasks because semantic relations rely heavily on wider contexts and stronger semantic comprehension of the sentence. In our experiments, we formulate the RC task as a supervised classification problem and learn a hypothesis g, which assigns a class y to given input sentence s with two entities e_1 and e_2, *i.e.*, $g(s, e_1, e_2) = y$. In general, a sentence can often

have multiple entities and multiple relations among them. In such cases, separate instances will be created for each pair of entities, where sentence will remain the same but target entities will differ. In this manner, model can classify relations between all pairs of entities in a given sentence.

1.2 Standard Methods

Named Entity Recognition

Named entity recognition is a well-studied area of research in NLP. Existing methods can be broadly categorized in three ways : dictionary-based, rule-based and learning-based methods. The dictionary-based approach employs the help of certain lexical resources whose vocabulary entities of interest such as drugs and diseases [Aronson, 2001]. The rule-based method uses morphological and orthographic patterns to predict entities in the document [Farmakiotou et al., 2000]. Exact or partial matching with different rules is generally used to determine whether classifying a word should be classified as an entity. Complicated constructions, irregular naming conventions, cascaded named entities, nonstandard abbreviations and the increasing number of biomedical entities make the rules and dictionary-based methods poor performers for NER tasks. Furthermore, neither method is generalizable for each new entity type; it would require either specific lexical resources or a new set of rules suitable for that specific entity type.

In contrast, the learning-based method uses a large annotated dataset and a machine learning tool to identify entities in the text [Settles, 2005, Leaman and Gonzalez, 2008,Leaman et al., 2009]. Most of the learning-based methods formulate NER task as a sequence labeling task. To train an appropriate classifier, methods, falling into this category, firstly, represent each word as a vector by generating thousands of task-specific features. In the next step, a classifier is used to model

a probability distribution over all possible labels or tags. Hidden Markov Models (HMM) [Rabiner, 1989], Maximum Entropy Markov Models (MEMM) [McCallum et al., 2000], Conditional Random Fields (CRF) [Lafferty et al., 2001] and Support Vector Machines (SVM) [Cortes and Vapnik, 1995] are the most widely used machine learning models in this category. HMM, being a generative model, estimates the joint probability between word and label sequences. The estimated joint probability is then used for prediction of a label sequence for a given sentence. HMM uses n-gram statistics of the training dataset to estimate model parameters. MEMM is a discriminative model and thus estimates conditional probability in contrast to the joint probability used in HMM. MEMM is a conditional tagging model which uses a log-linear model for modeling a distribution over all possible tags of a word. The key advantage of MEMM over HMM is that it allows a model to incorporate a flexible set of features. In contrast to MEMM, CRF uses the conditional probability distribution of a whole sequence to predict the correct label sequence of an input sequence. SVM is another widely used machine learning model for the entity recognition task. High levels of performance have also been achieved through this method also [Li, 2012, Björne et al., 2013]. SVM can be effectively used for nonlinear classification tasks [Cristianini and Shawe-Taylor, 2000] with the help of kernel trick [Milanfar, 2013]. In contrast to CRF, SVM does not use label dependency information.

Relation Classification

Relation classification in unstructured text has been modeled in many different ways. *Co-occurrence* based methods are the most widely used methods in the biomedical and clinical domain due to their simplicity and flexibility. In co-occurrence analysis, it is assumed that if two entities occur together in many sentences, there must be a relation between them [Bunescu et al., 2006, Song et al., 2011]. However, these methods cannot differentiate between different types of relations and suffer

from low precision and recall. Different statistical measures such as pointwise mutual information, chi-square or log-likelihood ratio have been used in these approaches [Stapley and Benoit, 2000] to improve the performance. *Rule-based methods* are another commonly employed method for the relation classification task [Thomas et al., 2000, Park et al., 2001, Leroy et al., 2003]. Rules are created by carefully observing the syntactic and semantic patterns of relation instances. The *bootstrapping method* [Xu, 2008] is used to improve the performance of rule-based methods. Bootstrapping uses a small number of known relation pairs of each relation type as seeds and uses these seeds to search patterns in a huge amount of unannotated text in an iterative fashion [Xu, 2008]. The bootstrapping method also generates many irrelevant patterns, which can be controlled by the *distant supervision* approach. A distantly supervised method uses a large knowledge base such as UMLS or Freebase as its input and extracts patterns from a large corpus for all pairs of relations present in the knowledge base [Mintz et al., 2009, Riedel et al., 2010, Roller and Stevenson, 2014]. The advantage of bootstrapping and distantly supervised methods over supervised methods is that they do not require large amounts of manually labeled training data which is typically hard to obtain.

Feature-based methods use sentences with predefined entities to construct a feature vector through feature extraction [Hong, 2005, Minard et al., 2011, Rink et al., 2011]. Feature extraction is mainly based on the output of linguistic and domain-specific tools. The extracted feature vectors are used to determine the correct class of relation using a given classification techniques. *Kernel methods* are an extension of feature-based methods which utilize kernel functions to exploit rich syntactic information such as parse trees [Zelenko et al., 2003, Culotta and Sorensen, 2004, Qian and Zhou, 2012, Zeng et al., 2014]. State-of-the-art results have been obtained by this class of methods. However, the performance of feature and kernel-based methods is highly dependent on suitable feature set selection, which

is not only tedious and time-consuming task, but also requires domain knowledge and is dependent on other NLP systems. Often, such dependencies reduce the reproducibility of existing work simply because of the absence of the full and finer details of feature extraction. Often, these methods result in a huge number of features and may be affected by the curse of dimensionality issues [Bengio et al., 2003, Collobert et al., 2011]. Another issue faced by these methods is that feature extraction needs to be adjusted according to the data source.

1.3 Contribution of the Thesis

In this thesis, we explore several neural network architectures for extracting entities and relationships in biomedical and clinical text. Our main objective is to exploit the power of representation learning and use minimal feature engineering in designing models. The main contributions of the thesis are summarized below.

Named Entity Recognition

Most existing methods for the biomedical entity recognition task rely on manually designed features which generally corresponds to the output of other existing NLP tools and are specific to a particular task of interest. We propose a unified framework for different biomedical and clinical entity recognition tasks. The model is unified in the sense that it does not require any domain or task-specific features. In other words, even though the nature and the source of texts vary for biomedical and clinical entity recognition tasks, the unified model uses the same feature types. The proposed unified framework uses two different bi-directional long short-term memory networks (BLSTMs) in a hierarchy for learning morphologically and contextually rich feature vectors. Similarly to other existing sequence labeling models, our model uses the conditional random field (CRF) in a cascade of BLSTM to infer a label sequence

from feature vectors. The use of the CRF layer in the model allows it to include label dependencies. In our analysis, we also use other features such as PoS tags. All of these features are also embedded in $\mathbb{R}^d$. The objective of this analysis is to explore the effectiveness of representation learning with and without the use of additional features that are dependent on other tools and may also require specific training datasets. By carrying out the experiments on three NER tasks, namely, disease NER, drug NER and clinical NER, we demonstrate that state-of-art results can be achieved with representation learning and minimal feature engineering. *This chapter is based on the papers [Sahu and Anand, 2016] and [Sahu and Anand, 2017b].*

Convolution Neural Network for Relation Classification

Relation classification is the process of detecting and classifying the semantic relation present between entities in a given text. Here, we assume that all the interested entities are given, and our task is to determine which class of relations holds between the target entities. Existing models for this task use either manually engineered features or kernel methods to create a feature vector. These features are then fed to a classifier to predict the correct class. The results of these methods are highly dependent on the quality of user-designed features and they also suffer from the curse of dimensionality. In this work, we focus on extracting relations from clinical discharge summaries and MedLine abstracts. Our primary objective is to exploit the power of CNN to learn features automatically and thus reduce to the dependency on manual feature engineering. We evaluate the performance of the proposed model on the I2B2/VA-2010 clinical relation classification and SemEval 2013 DDI extraction challenge datasets. Our results indicate that CNN is an effective model for relation exaction in the biomedical and clinical text without being dependent on manual feature engineering. *This chapter is based on the paper [Sahu et al., 2016].*

LSTM Network for Relation Classification

CNN is a powerful model to learn features from continuous n-grams in the sentence. However, it may cause problems for long sentences or those having important clues lying far away from each other. In this work, we present three models, namely, *BLSTM-RE*, *ABLSTM-RE* and *Joint ABLSTM-RE*, based on the LSTM network. All three models utilize word and position embedding as latent features and thus do not rely on feature engineering. Further use of BLSTM networks allows extraction of features from the whole sentence. The two models, *ABLSTM-RE* and *Joint ABLSTM-RE* also use attentive pooling in the output of BLSTM layer to assign weights to the features. We evaluate the performance of the proposed models on clinical relation classification and drug-drug interaction extraction tasks. Our results indicate that the proposed model achieves reasonable performance on both the datasets. Analysis of our results indicates that LSTM based models also find it difficult to make correct predictions for entity pairs in long sentences that contain many other entities. However, if we compare CNN and LSTM based models, LSTM models are generally found to be more accurate for longer sentences than the CNN model. *This chapter builds on the paper [Sahu and Anand, 2017a] and partially [Raj et al., 2017].*

Transfer Learning for Relation Classification

The lack of sufficient labeled data often limits the ability to apply advanced machine learning algorithms to real-time problems. However, efficient use of *Transfer Learning* (TL) has been shown to be very useful across domains. TL utilizes valuable knowledge learned in one task (*source task*), where sufficient data is available, to a task of interest in another domain (*target task*). In the biomedical and clinical domains, it is quite common that a lack of sufficient training data restrict the potential to fully exploit machine learning models. In this work, we present two

unified recurrent neural models, leading to three transfer learning frameworks for relation classification tasks. We systematically investigate the effectiveness of the proposed frameworks in transferring the knowledge under multiple aspects related to the source and target tasks, such as similarity or relatedness between the two tasks, and the size of the training data for the source task. Our empirical results show that the proposed frameworks, in general, improve the model performance. However, these improvements do depend on different characteristics of the source and target tasks. These dependencies, determine the choice of most appropriate TL framework. *This chapter is based on the [Sahu and Anand, 2018] paper.*

1.4 Outline of the Thesis

Briefly, the thesis is organized as follows. We begin by giving a brief description of various neural networks and word embedding techniques in **Chapter 2**. Different biomedical and clinical entity recognition tasks are described in **Chapter 3**. **Chapter 4** and **Chapter 5** explore CNN and LSTM networks for the two RE tasks, namely drug-drug interaction and clinical relation classification tasks, respectively. In **Chapter 6**, we present various transfer learning techniques which utilize the knowledge learned on another for different relation classification tasks. **Chapter 7** concludes the thesis and outlines future works.

Chapter 2

Background

2.1 Overview

This chapter briefly describes all relevant neural networks used in our proposed models and their training mechanisms. We also summarize two word embedding techniques, namely, *word2vec* and *GloVe*. In this thesis, we used *GloVe* to obtain word vectors. However, we observe that *word2vec* also gives similar performance. Finally, we conclude this chapter by reporting about the corpus and preprocessing used to obtain the pre-trained word vectors used in our experiments.

2.2 Neural Networks

A neural network or artificial neural network is a class of learning models. It was originally inspired by the network of biological neurons, such as those found in the brain. In this section, we give a brief description of the different neural networks used in the thesis.

Single Neuron

A neuron is a computational unit that takes as its input a vector and computes a real number as an output, with the help of an activation function. A typical neural network can have multiple neurons in a hierarchy. Each neuron has a set of parameters, which are updated during the training.

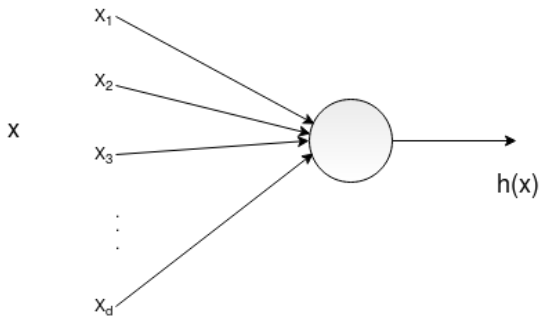


Figure 2.1: Example of single neuron

Let $x \in \mathbb{R}^d$ be the input for the neuron h and weight, $w \in \mathbb{R}^d$ and bias, $b \in \mathbb{R}$ are the parameters of the neuron, then activation of h would be:

$$h(x) = f(x \cdot w + b)$$

Here, f is an activation function and $\cdot$ indicates the dot product. Typically, *sigmoid* or *tanh* are used as the activation function. A single neuron can be used to model a binary classifier, a linear and a nonlinear regression.

Feedforward Neural Network

The feedforward neural network is the simplest form of neural network. It takes inputs in the first layer of the network and produces output at the final layer. All the computations happen in a forward direction in a layer wise fashion. A typical architecture of a three layered feedforward neural network is presented in Figure 2.2.

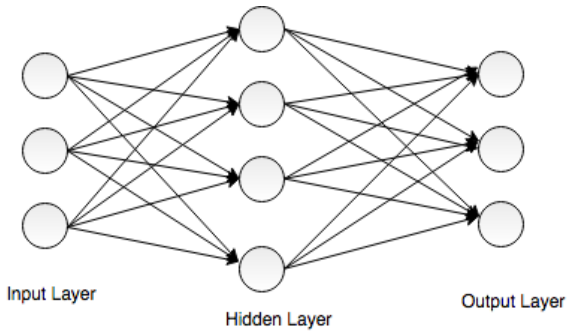


Figure 2.2: Example of a feed forward neural network

Let $x \in \mathbb{R}^{d_1}$ be the input for the neural network and $W_1 \in \mathbb{R}^{d_1 \times d_2}$, $b_1 \in \mathbb{R}^{d_2}$ are the parameters of the first layers' neurons, then the activations of first hidden layer would be:

$$z^{(2)} = f(x \cdot W_1 + b_1)$$

Here, the activation function f is applied to the vector in an element-wise fashion. In our subsequent discussions, we always assume that a function is applied to a vector in an element-wise fashion, unless otherwise stated. The final output of the entire network would be:

$$z^{(3)} = f(z^{(2)} \cdot W_2 + b_2)$$

where $W_2 \in \mathbb{R}^{d_2 \times d_3}$ and $b_2 \in \mathbb{R}^{d_3}$ are the parameters of the output layer's neurons. A feedforward neural network can be used for multiclass classification problems. In typical architecture, the number of neurons in the final layer is equal to the number classes. A feedforward neural network is also called *fully-connected neural network*, as all the nodes of the previous layer connect with each node of next layer.

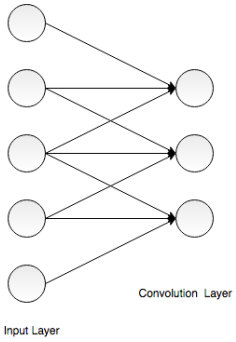


Figure 2.3: Convolution neural network

Convolution Neural Network

The convolution neural network (CNN) is a kind of feedforward neural network, which learns local features by restricting the hidden layer neurons to take their input from a part of the input layer or previous hidden layer neurons (as presented in Figure 2.3) [LeCun et al., 1998, Krizhevsky et al., 2012]. The motivation behind applying CNN to text is to learn local features, irrespective of their positions. CNN has been successfully applied to several NLP problems, such as PoS tagging, chunking and semantic role labeling [Collobert and Weston, 2008], text classification [Kim, 2014, Kalchbrenner et al., 2014, Zhang et al., 2015] etc.

A typical CNN architecture consists of a *convolution layer*, a *pooling layer* and a *fully-connected layer*. A *convolution layer* has a set of learnable filters. Each filter is a small vector. During the forward pass, we slide (convolve) each filter across the length of the input sequence and compute dot products between the entries of the filter and the input at any position. As we slide the filter over the length of the input sequence, we produce an activation map that provides the responses of that filter at every spatial position. Intuitively, the network will learn filters that activate when they see some clue that is important for the task. In experiments, we generally use several such filters with different lengths. The output of the *convolution layer* is a

set of activation maps for each filter over the input sequence. We use the *pooling layer* after *convolution layer* to aggregate the learned features of the convolution layer. In general, we use max pooling in the *pooling layer*. It only considers the most useful features from the entire sequence.

Let $x_1, x_2 \dots x_m$ be the sequence of input vectors, where $x_j \in \mathbb{R}^d$ is the vector for $1 \leq j \leq m$ and m is the length of the sequence. Let $w^i \in \mathbb{R}^{d \cdot c}$ and $b^i \in \mathbb{R}$ be the parameters of CNN, where c is the length of filters. Also let $x_{j:j+c}$ represent the concatenation of $x_j \dots x_{j+c}$ input vectors. Then the convolution layer and max pooling layer compute outputs as follows:

$$h_j^i = f(w^i \cdot x_{j:j+c-1} + b^i) \qquad for\ \ j = 1, 2, \dots m - c + 1$$
$$z^i = \max_{1 \leq j \leq (m-c+1)} [h_j^i]$$

where f is the activation function, w^i and b^i are the learning parameters that will remain same for all $i = 1, 2, \dots m - c + 1$. In this computation, we used one filter with length equals to c. Similar computations can be applied for all such filters.

Recurrent Neural Network

Feedforward neural networks are unable to take into account any dependencies present in sequential data. However, in many NLP tasks, we need to accommodate sequential information for learning appropriate feature representations. The recurrent neural network (RNN) is a particular kind of network which maintains a loop in the computation of its output. A loop allows information to be passed from one step of the network to the next [Bengio et al., 1994, Graves, 2013]. At any time step t, RNN takes as input a vector $x_t \in \mathbb{R}^d$ and produces as output a vector $h_t \in \mathbb{R}^k$. For the computation of output h_t, RNN uses inputs x_t as well as the state of the output at $(t-1)$. Let $U \in \mathbb{R}^{k \times d}$, $W \in \mathbb{R}^{k \times d}$ and $b \in \mathbb{R}^k$ be the parameters of RNN and let h_{t-1} be the output of RNN at time step $(t-1)$, then the computation

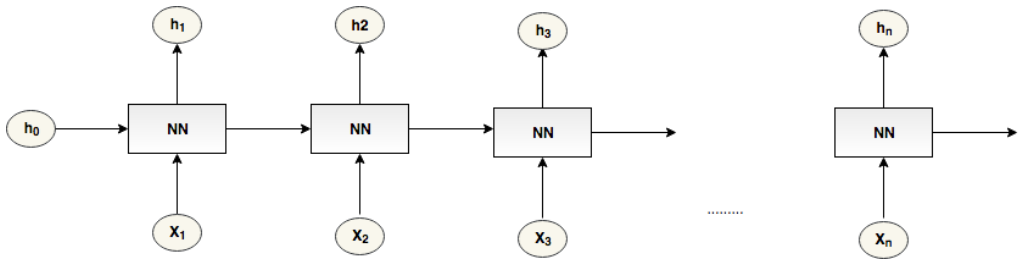


Figure 2.4: Block digram of RNN

of h_t would be:

$$h_t = tanh(W \cdot x_t + U \cdot h_{t-1} + b)$$

Here, h_0 is also a learning parameter, which is initialized with random values. One can think of the hidden state h_t as the memory of the network. h_t captures information about what happened in all of the previous time steps. The output at step t is calculated solely based on the memory at time t. It is some what more complicated in practice, because h_t typically cannot capture information from more distinct time steps.

Similarly, to a neural network, RNN requires training through the backpropagation algorithm (discussed in section 2.3). Therefore, while training, it will unfold the RNN to make it a feedforward neural network. When the gradient is passed back through many time steps, it tends to grow or vanish, similarly to a multi layer neural network [Pascanu et al., 2012]. This makes the training of RNN difficult in real time.

Long Short-Term Memory Network

Traditional RNN models suffer from both vanishing and exploding gradient [Pascanu et al., 2012, Bengio et al., 2013]. Such models are likely to fail where longer contexts are required to carry out tasks effectively. These issues were the primary motivation

behind the development of the long short-term memory (LSTM) model [Hochreiter and Schmidhuber, 1997]. The LSTM layer is another way to compute hidden states which introduces a new structure called a memory cell (c_t) and three gates called input (i_t), output (o_t) and forget (f_t) gates. These gates are composed of sigmoid activation functions and the responsible for regulating information in the memory cell. The input gate, by allowing an incoming signal to alter the state of the memory cell, regulates the proportion of history information that the memory cell will keep. On the other hand, the forget gate can modulate the memory cells and allows the cell to remember or forget its previous state. Finally, the output gate regulates the proportion of stored information in the memory cell that will influence the output. The computation of memory cell (c_t) is carried out using previous states of the memory cell and candidate hidden state (g_t) which we compute using the current input and the previous hidden state. The final output of a hidden state is calculated based on the current status of the memory cell and the output gate.

Let x_t be the input vector for t^{th} element in a sequence and h_{t-1} be the previous hidden state, then computation of LSTM output (h_t) would be:

$$
\begin{aligned}
i_t &= \sigma(U_i \cdot x_t + W_i \cdot h_{t-1} + b_i) \\
f_t &= \sigma(U_f \cdot x_t + W_f \cdot h_{t-1} + b_f) \\
o_t &= \sigma(U_o \cdot x_t + W_o \cdot h_{t-1} + b_o) \\
g_t &= tanh(U_g \cdot x_t + W_g \cdot h_{t-1} + b_g) \\
c_t &= c_{t-1} * f_t + g_t * i_t \\
h_t &= tanh(c_t) * o_t
\end{aligned}
$$

where σ is a sigmoid activation function, $*$ is an element wise product, U_i, U_f, U_o, $U_g \in \mathbb{R}^{N \times d}$, W_i, W_o, W_f, $W_g \in \mathbb{R}^{N \times N}$ and b_i, b_f, b_o, $b_g \in \mathbb{R}^N$, h_0, $c_0 \in \mathbb{R}^N$ are learning parameters for the LSTM. Here, d is the dimension of input feature vector, N is the hidden layer size and h_t is the output of LSTM at time step t.

It has become common practice to use LSTM in both forward and backward directions to capture both past and future contexts, respectively. Firstly, LSTM computes its hidden states by moving in a forward direction over the input sequence and secondly it moves in a backwards direction. This way of using two LSTMs is referred to as bi-directional LSTM or simply BLSTM. The final output of BLSTM at time t is given as:

$$h^{(t)} = \overrightarrow{h_t} \oplus \overleftarrow{h_t} \tag{2.1}$$

where $\oplus$ is the concatenation operation and $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ are hidden states of forward and backward LSTM at time t.

2.3 Training of Neural Network

A neural network based model can be trained in several ways depending upon the task and architecture of the network. In this thesis, we use the neural network as a *supervised learning* algorithm, which assumes that we have a training dataset for which desired output is given. A supervised model creates a hypotheses $g : (x, w) \rightarrow y$, here $x \in X$ is the input sample, $y \in Y$ is the desired output for x and w is the set of learning parameters of the model. In the classification problem, Y is a set of discrete values.

2.3.1 Cross Entropy Loss

For training a neural network based model, we need to maintain a *loss function* that measures the quality of a particular set of learning parameters based on how well the induced scores agree with the ground truth labels in the training data. In our experiments, we use a cross entropy loss function over *softmax classifier* for training our models. At any time, softmax takes as input an unnormalized vector and produces a normalized vector as output. Therefore, the cross entropy loss for

an input instance i would be:

$$L_i = -\log\left(\frac{e^{o^{(i)}_{y_i}}}{\sum_{\forall j} e^{o^{(i)}_j}}\right) \tag{2.2}$$

Here y_i is correct relation class for i^{th} instance and $o^{(i)}$ is the input vector for the cross entropy loss function. In our models, $o^{(i)}$ is the output of final layer of neural network models.

2.3.2 Backpropagation Algorithm

Like other machine learning based models, training a neural network also needs to compute partial derivative (gradient) of loss function with respect to all the learning parameters of the model. Because of the recursive nature of computation in neural network (the output of a layer would be the input for next layer), it is not possible to obtain the gradient of loss function in a classic manner. Backpropagation is a way to compute the gradient for a loss function based on a recursive application of the *chain rule* [Rumelhart et al., 1988].

A neural network based model can be seen as a composition of N recursive functions or layers $f^l_\Theta(.)$. Each layer consist of a parameter matrix Θ which need to be updated while training:

$$f_\Theta(.) = f^{(N)}(..f^{(2)}(f^{(1)}(x))...) \tag{2.3}$$

Let $\Theta^1, \cdots \Theta^l \cdots \Theta^N$ be the parameters of N layers in the network. We need to compute the gradients of the loss function with respect to each Θ^l. If Ψ is the loss function of the model then, by using chain rule we can obtain the classical backpropagation recursion as:

$$\frac{\partial \Psi}{\partial \Theta^l} = \frac{\partial f^{l-1}_\Theta}{\partial \Theta^l} * \frac{\partial \Psi}{\partial f^{l-1}_\Theta} \tag{2.4}$$

$$\frac{\partial \Psi}{\partial f^{l-1}_\Theta} = \frac{\partial f^{l}_\Theta}{\partial f^{l-1}_\Theta} * \frac{\partial \Psi}{\partial f^{l}_\Theta} \tag{2.5}$$

2.3.3 Parameter Update

Once we have computed the gradients for all parameters, they are used to perform updates to the parameters. There are various approaches for performing the update, which we discuss below. Suppose the vector of a parameter that we want to update is w and its gradient is dw:

1. **Vanilla update**: The simplest and most common form of update is to change the parameters opposite to their gradient directions (since the gradient indicates the direction of increase, but we usually need to minimize a loss function).

   $$w = w - \eta * dw$$

   where η is a fixed constant, called *learning rate*. When evaluated on the complete dataset, and when the learning rate is small enough, this is guaranteed to make non-negative progress on the loss function.

2. **Momentum update**: Momentum update is another widely used optimization technique. While updating, it assigns weights to the past gradients of the parameters. It is used to diminish the fluctuations in weight changes over consecutive iterations. It maintains a velocity vector v and a hyperparameter μ called momentum. The size of v is the same as parameter vector w, and it is initialized with zeros.

   $$v = \mu * v - \eta * dw$$
   $$w = w + v$$

3. **Adam**: In our all experiments we use the *adam* optimization technique for training the models. The name *adam* is derived from adaptive moment estimation [Kingma and Ba, 2014]. Adam combines the properties of the

AdaGrad [Duchi et al., 2011] and RMSProp[1] algorithms to provide an optimization algorithm that can handle sparse gradients on noisy problems. It is well suited to problems that are large in terms of data or parameters. Adam's hyperparameters have intuitive interpretation and typically require little tuning. The method computes individual adaptive learning rates for different parameters from estimates of the first and second moments of the gradients.

$$m = \beta_1 * m + (1 - \beta_1) * dw$$
$$v = \beta_2 * v + (1 - \beta_2) * (dw)^2$$
$$w = w - \eta * m/(np.sqrt(v) + \epsilon)$$

Notice that the variables v and m have sizes equal to the size of the gradients and they keep track of the per-parameter sum of gradient and the sum of squared gradients with momentum, respectively. This is then used to normalize the parameter update step, element-wise. Recommended values for hyperparameters in the paper [Kingma and Ba, 2014] are ϵ = 1e-8, β_1 = 0.9, β_2 = 0.999

2.4 Word Embedding

One of the crucial steps in machine learning (ML) based NLP models is to produce an appropriate representation of words as input to a model. Initially most work treated each word as an atomic symbol and assigned a one hot vector to each word [Leaman et al., 2009, Ko, 2012]. The length of the vector in this representation is equal to the size of the vocabulary and the element at the word index is 1, while the other elements are 0s. The two major drawbacks with this representation are:

[1] `http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf`

firstly, length of the vector is enormous, and secondly, there is no notion of similarity between words. The inability of the one-hot vector representation to embody lexico-semantic properties prompted researchers to develop methods which are based on the idea that the "similar words appear in similar contexts" [Lin et al., 2003]. These methods can be broadly classified into two categories [Turian et al., 2010], namely, *distributional representation* and *distributed representation*. Both groups of methods work in an unsupervised manner and utilize a massive corpus to obtain a vector representation or embedding. Distributional representations are mainly based on a co-occurrence matrix O of words in the vocabulary and their contexts. Here, among other possibilities, contexts can be documents or words within a particular window. Each entry O_{ij} in the matrix may indicate either the frequency of word i in the context j or simply whether the word i has appeared in the context j at least once. Co-occurrence matrices can be designed in a variety of ways, as summarized by [Turney and Pantel, 2010, Jurafsky, 2000] has discussed several of them. The major issue with such methods is the huge size of the matrix O, and reducing its size tends to be computationally very expensive. Nevertheless, the requirement of constructing and storing O is always there. The second group of methods is mainly based on language modeling [Bengio et al., 2003] which we used in all our experiments.

Outside the biomedical domain, word embeddings have resulted significant improvements in the performance of many NLP tasks. For example, [Turian et al., 2010] improved the performance of chunking and named entity recognition by using word embedding as one of the features in their CRF model. Collobert and his colleagues in [Collobert et al., 2011] formulated the NLP tasks of PoS tagging, chunking, named entity recognition and semantic role labeling as a multi-task learning problems. They showed improvement in the performance when word vectors are learned together with other NLP tasks. Socher and co-authors in [Socher et al., 2012] improved the performance of sentiment analysis and a semantic relation

classification task using the recursive neural network. The common step among these models is: learning of word embeddings from a huge unannotated corpus like Wikipedia. The embedding were later used as a features in a variety of ways. We next discuss the two most commonly used word embedding techniques, which we used for learning word vectors in our experiments.

2.4.1 word2vec

word2vec is an efficient way of training neural network based word embedding as it does not involve dense matrix multiplication [Mikolov et al., 2013b, Mikolov et al., 2013a]. *word2vec* has two variants of the language model, called *Continuous Bag of Words* (CBOW) and *Skip-gram*. Both of these models take huge corpus as an input and learn two vector representations (input and output vector representation) for each word in it. The final vector representation is simply the average of the two learned vector representations. Below we briefly describe the two models.

1. **CBOW:** CBOW takes a window of preceding and following words its input and predict the center word as its output. Let u_w and v_w denote the input and output vector of word w, W, the number of words we have in vocabulary and let c denote the context window size, then in CBOW the objective function we need to maximize is :

$$\frac{1}{T}\sum_{t=1}^{T}\log p(w_t|w_{(t-c)},..w_{(t-1)},w_{(t+1)},...w_{(t+c)}) = \frac{1}{T}\sum_{t=1}^{T}\log\frac{\exp(h_t^T v_{w_t})}{\sum_{w=1}^{W}\exp(h_t^T v_w)}$$

   where

$$h_t = \frac{1}{2c} \sum_{-c \leq j \leq c, j \neq 0} u_{(t+j)}$$

2. **Skip-gram:** Skip-gram is the mirror image of CBOW. It takes one word as its input and predicts the whole context of words as its output. The objective function for skip-gram also uses softmax :

$$\frac{1}{T} \sum_{t=1}^{T} \log p(w_{(t-c)}, ..w_{(t-1)}, w_{(t+1)}, ...w_{(t+c)} | w_t)$$
$$= \frac{1}{T} \sum_{t=1}^{T} \log \prod_{j=0, j \neq c}^{2c} p(w^{(t-c+j)} | w^{(t)})$$

where

$$p(w^{(t-c+j)} | w^{(t)}) = \frac{\exp(v^T_{w^{(t-c+j)}} u_{w_t})}{\sum_{w=1}^{W} \exp(v^T_w u_{w_t})}$$

Since W is a huge number in any corpus, the author used hierarchical softmax [Morin and Bengio, 2005] and negative sampling [Collobert and Weston, 2008] approaches to approximate these objective functions. For our experiments, we downloaded *word2vec* tool[2] which is efficiently implemented in the C language.

2.4.2 GloVe

GloVe (Global Vector) [Pennington et al., 2014] combines the properties of distributional semantics as well as continuous word vector models. It also maintains two vectors for each word, one for the word itself and the other for the context of the word. *GloVe* tries to learn vectors for words w_x and w_y such that their dot product is proportional to their co-occurrence count. Let u_i and v_i denote the word vector

[2] `https://code.google.com/p/word2vec/`

and context vector, respectively, for the word w_i, $O_{i,j}$ is the co-occurrence count of words w_i and w_j and V is the vocabulary size, then objective function of *GloVe* is calculated as:

$$J = \sum_{i,j=1}^{V} f(O_{(i,j)})(u_i^T v_j + b_i + d_j - \log X_{i,j})^2 \tag{2.6}$$

where b_i and d_i denote the biases for word w_i and *f(x)* is a weighting function. *f(x)* takes the value zero if x is zero and one if x is larger than a number x_{max}.

$$f(x) = \begin{cases} (\frac{x}{x_{max}})^{\alpha} & \text{if } x < x_{max} \\ 1 & \text{otherwise.} \end{cases}$$

In the experiments, the authors used $x_{max} = 100$ and $\alpha = \frac{3}{4}$. It should be noted here that when the co-occurrence count between the words is zero, the weighting function will also return zero. Therefore, its complexity is not high. For our experiments, we used the *GloVe* tool[3] which is also implemented in C language.

2.4.3 Corpus Data and Preprocessing

PubMed Central® (PMC) is a repository of biomedical and life sciences journal literature at the U.S. National Institutes of Health's National Library of Medicine (NIH/NLM). The PMC ftp service[4] provides access to source files for articles in the *Open Access Subset* in variety of ways. We downloaded the gzipped archived files of full-length texts of all articles in the open access subset on 19th April 2015. This corpus contains around 1.25 million articles having a total of around 400 million tokens.

To pre-process the corpus, we performed the following operations-

- We used Stanford segmenter and tokenizer[5] for sentence segmentation and word tokenization respectively.

[3] `http://nlp.stanford.edu/projects/glove/`

[4] `http://www.ncbi.nlm.nih.gov/pmc/tools/ftp/`

[5] `https://nlp.stanford.edu/software/tokenizer.html`

- All the words in our corpus are transformed to lower case.
- All numbers are converted to a special symbol DG.
- Words occurring less than 50 times in the corpus are converted to a special token $< RARE >$.

2.5 Evaluation Metrics of NER and RE

For the evaluation of our model's performance on the NER and RE tasks, we use *Precision*, *Recall* and *F Score* measures, which depend on the following counts:

- **True positive (TP):** Number of samples which are correctly classified by the model.
- **False positive (FP):** Number of samples which are incorrectly classified by the model.
- **False negative (FN):** Number of samples which the model fails to classified.

Given that TP, FP and FN for the model we compute precision, recall and F score as follows:

$$Precision = \frac{TP}{(TP + FP)}$$

$$Recall = \frac{TP}{(TP + FN)}$$

$$FScore = \frac{2 * Precision * Recall}{(Precision + Recall)}$$

In multi-class classification problems, we used *micro* score for each evaluation metrics. Micro score calculates the metrics globally by counting the total true positives, false negatives and false positives [Tjong Kim Sang and De Meulder, 2003a].

2.6 Conclusion

In this chapter, we firstly briefly described different neural networks used in this thesis and their training strategies. Next, we explained different word embedding techniques *i.e word2vec* and *GloVe*. Finally, we introduced the evaluation scheme used for NER and RE tasks.

Chapter 3

Named Entity Recognition

3.1 Overview

Named entity recognition is one of the necessary steps in several biomedical and clinical information extraction tasks. Most existing methods for this task rely on explicit feature engineering, where many features are either specific to a particular task or depend on the output of other existing NLP tools. In this chapter, we propose a unified framework using BLSTM networks for the named entity recognition task in the biomedical and clinical domains. Three important characteristics of the framework are as follows - (1) the model learns contextual as well as morphological features using two different BLSTMs, (2) the model uses a first order linear CRF in its output layer in cascade of BLSTMs to a infer label sequence, (3) the model does not use any domain specific features or dictionary, in other words, the same set of features is used in the three different NER tasks, namely, disease NER, drug NER and clinical NER. We compare the performance of the proposed model with that of existing state-of-the-art models on standard benchmark datasets for the three tasks. We show empirically that the proposed framework outperforms all existing models. Furthermore, our analysis of the CRF layer and word-embedding obtained using

character-based embedding demonstrates their importance.

3.2 Introduction

Biomedical and clinical named entity recognition is an essential step in several biomedical and clinical information extraction tasks [Rosario and Hearst, 2004, Segura-Bedmar et al., 2015, Uzuner et al., 2011]. State-of-the-art methods formulate NER task as a sequence labeling problem where each word is labeled with a tag and based on the tag sequence, entities of interest are identified. It has been observed that named entity recognition in the biomedical and clinical domains is more difficult than the general domain [Leaman et al., 2009, Uzuner et al., 2011]. There are several reasons for this, including the use of non-standard abbreviations or acronyms, multiple ways of referring the same concepts, etc. Furthermore, clinical notes are noisier, grammatically error prone and contain less context than standard text due to shorter and incomplete sentences [Uzuner et al., 2011]. The most widely used models, such as maximum entropy markov models (MEMMs), CRFs, and support vector machines (SVMs), use manually designed features to obtain morphological, syntactic, semantic and contextual information for a word or of a piece of text surrounding a word, and use them as features for determining the correct label [Lafferty et al., 2001, Mahbub Chowdhury and Lavelli, 2010, Jiang et al., 2011, Rocktäschel et al., 2013, Björne et al., 2013]. It has been observed that the performance of such models is limited by the choice of explicitly designed features, which are generally specific to the task and its corresponding domain. For example, Chowdhury and Lavelli explained several reasons why features designed for biological entities such as protein or gene are not equally important for disease name recognition [Mahbub Chowdhury and Lavelli, 2010].

Deep learning based models have been used to reduce manual efforts for explicit feature design in [Collobert et al., 2011]. Here, distributional features

are used in place of manually designed features and a multilayer neural network is used in place of a linear model to overcome the needs of meticulous task-specific feature engineering. Although the proposed methods outperformed several generic domain sequence tagging tasks, their performance are fail to surpass the state-of-art in the biomedical domain [Yao et al., 2015]. There are two plausible reasons for this. Firstly, the model learned features only from a word level embedding and secondly, it took into account only a fixed length context around the word. It has been observed that word level embeddings preserve the syntactic and semantic properties of a word, but they may fail to maintain morphological information, which can also play an important role in biomedical entity recognition [dos Santos and Zadrozny, 2014, Lample et al., 2016, Mahbub Chowdhury and Lavelli, 2010, Leaman and Gonzalez, 2008]. For instance, the drug names *Cefaclor, Cefdinir, Cefixime, Cefprozil, Cephalexin* have a common prefix and *Doxycycline, Minocycline, Tetracycline* have a common suffix. Furthermore, a window based neural architecture can only consider contexts falling within the user determined window size and will fail to pick up on important clues lying outside the window.

This work aims to overcome the two issues mentioned above by using two BLSTMs in a hierarchy. The first BLSTM works on character vectors of a word to obtain morphologically rich word embeddings. The second BLSTM works on the word vectors of a sentence to learn contextually rich feature vectors. As a result, each feature vector will preserve morphological as well contextual features of a word within the sentence. The learned feature vectors are used in a CRF layer to predict the correct label sequence of a sentence. The CRF layer accommodates dependency information about the labels. We evaluate the proposed model on three standard biomedical entity recognition tasks, namely, disease NER, drug NER and clinical NER. To the best of our knowledge, this is the first work which explores the use of a single model using character-based word embedding in conjunction with word

embedding for both drug and clinical entity recognition tasks. We compare the proposed model with existing state-of-the-art models for each task and show that it outperforms them. Further analysis of the model indicates the importance of using character-based word embedding along with word embedding and the CRF layer in the final output layer.

3.3 Model Architecture

Similarly to all named entity recognition tasks, we formulate the biomedical entity recognition task as a token level sequence tagging problem. We use the BIO tagging scheme in our experiments [Settles, 2004]. The architecture of the proposed model is presented in Figure 3.1. Our model takes the whole sentence as its input and computes a label sequence as its output. The first layer of the model learns local feature vectors for each word in the sentence. We use the concatenation of word embedding, PoS tag embedding, and character-based word embedding as a local feature for every word. Character-based word embedding is learned through applying a BLSTM to the character vectors of a word. We call this layer *Char BLSTM* (Section 3.3.1). The subsequent layer, called *Word BLSTM* (Section 3.3.2), incorporates contextual information within it using a separate BLSTM network. Finally, we use a CRF layer (Section 3.3.3) to encode the correct label sequence using the output of *Word BLSTM*. Hence forth, the proposed framework will be referred to as *CWBLSTM*. The parameters of the entire network are trained in an end-to-end manner using a cross entropy loss function. Subsequently, we describe each part of the model in detail.

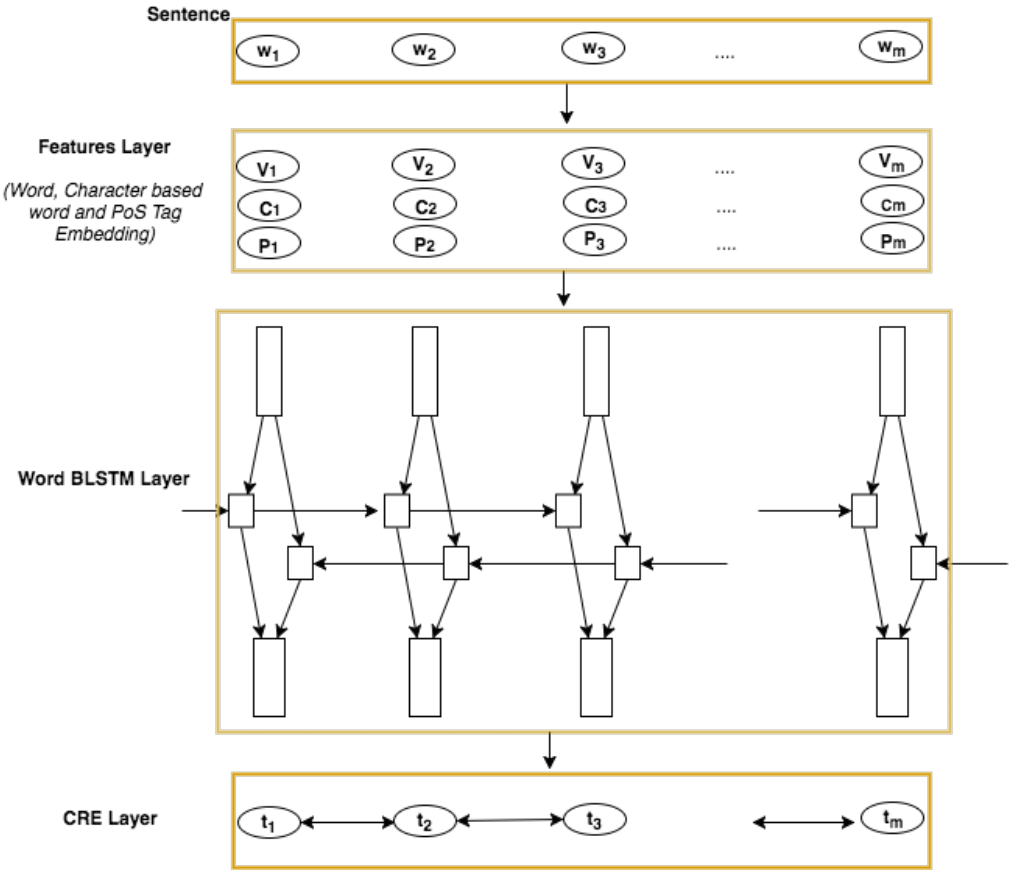


Figure 3.1: Architecture of CWBLSTM model. Here $w_1 w_2 ... w_m$ is the word sequence of the sentence and t_1 t_2 ... t_m is its computed label sequence and m represents length of the sentence.

3.3.1 Features Layer

Word embeddings or distributed word representations are a compact vector representations of words which preserve lexico-semantic properties [Bengio et al., 2003]. It is a common practice to initialize word embeddings with a pre-trained vector representation of words. In addition to word embeddings, PoS tags and character-based word embeddings are used as features. We use the GENIA[1] tagger to obtain PoS

[1]http://www.nactem.ac.uk/GENIA/tagger/

tags in all the datasets. Each PoS tag was initialized randomly and was updated during the training. The output of feature layer is a sequence of vectors $x_1, \cdots x_m$ for a sentence of length m. Here $x_i \in \mathbb{R}^d$ is the concatenation of word embeddings, PoS tag embeddings and character-based word embeddings. We next explain how character-based word embedding is learned.

Char BLSTM

Word embedding is a crucial component for all deep learning based NLP tasks. The capability to preserve lexico-semantic properties in the vector representation of a word makes it a powerful resource for NLP [Collobert et al., 2011, Turian et al., 2010]. In biomedical and clinical entity recognition tasks in addition to semantic information, morphological information such as prefix, suffix or certain standard patterns of words also provide important clues [Mahbub Chowdhury and Lavelli, 2010, Leaman et al., 2009]. The motivation behind using character-based word embeddings is to incorporate morphological information of words within feature vectors.

To learn character based embeddings, we maintain a vector for every character in an embedding matrix [dos Santos and Zadrozny, 2014,Lample et al., 2016]. These vectors are initialized with random values. As in example, suppose *cancer* is a word for which we want to learn an embedding. Figure 3.2 showed how we apply a BLSTM to the vector of characters in *cancer*. As mentioned earlier, the forward LSTM maintains information about past context in the computation of the current hidden state while the backward LSTM is used to obtain future contexts. Hence, after reading an entire sequence, the final hidden states of both LSTMs must have knowledge of the whole word with respect to their directions. The final embedding of a word is:

$$v_{cw} = \overrightarrow{h^{(m)}} \oplus \overleftarrow{h^{(m)}} \tag{3.1}$$

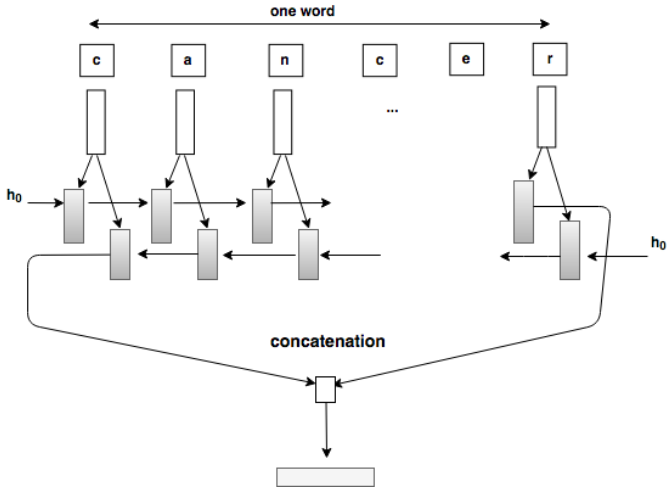


Figure 3.2: Learning character based word embedding

where $\overrightarrow{h^{(m)}}$ and $\overleftarrow{h^{(m)}}$ are the final hidden states of the forward and backward LSTMs respectively.

3.3.2 Word BLSTM Layer

The output of the feature layer is a sequence of vectors corresponding to the sequence of words in the sentence. These vectors have semantic information about the words. Although semantic information plays an important role in identifying entities, a word can have different meanings in different contexts. Earlier works *e.g.*, [Collobert et al., 2011, Leaman and Gonzalez, 2008, Mahbub Chowdhury and Lavelli, 2010, Yao et al., 2015] use a fixed length window to incorporate contextual information. However important clues can lie anywhere within the whole sentence. Thus fixed length window limits the ability of the learned vectors to obtain knowledge about the complete sentence. To overcome this, we use a separate BLSTM network which takes semantic vectors as input and outputs a vector for every word, based on both contexts and current feature vectors.

3.3.3 CRF Layer

The output of the *Word BLSTM* layer is again a sequence of vectors which have contextual as well as semantic information. One simple way to decode the feature vector of a word to determine its corresponding tag is to use word level log likelihood (WLL) [Collobert et al., 2011]. Similar to MEMMs, WLL will map the feature vector of a word to a score vector corresponding to each possible tag using a linear transformation. Each word is then assigned a label based on the score vector, independent of the labels of other words. The limitation of this decoding scheme lies in its inability to take into account dependencies among tags. For instance, in the *BIO tagging* scheme, a word can be tagged with *I-Entity* (standing for Intermediate-Entity) only after a *B-Entity* (standing for Beginning-Entity). We apply CRF [Lafferty et al., 2001] to the feature vectors to allow dependency information to be used in decoding, we decode the whole sentence together with its tag sequence.

CRF maintains two parameters for decoding, *i.e*, the linear mapping parameter $W_u \in R^{k \times h}$ and the pairwise transition score matrix $T \in R^{h \times h}$. Here, k is the length of the feature vector, h is the number of labels present in the task and $T_{i,j}$ implies the pair-wise transition score for moving from tag i to tag j. Let $[v]_1^{|s|}$ be a sequence of feature vectors for a sentence $[w]_1^{|s|}$ and suppose that $[z]_1^{|s|}$ are the unary potential scores obtained after applying a linear transformation on the feature vectors (here $z_i \in R^h$), CRE then decodes this with tag sequence using:

$$P([y]_1^{|s|}|[w]_1^{|s|}) = \underset{t \in Q^{|s|}}{\operatorname{argmax}} \frac{\exp \Psi([z]_1^{|s|}, [t]_1^{|s|})}{\sum_{t^\psi \in Q^{|s|}} \exp \Psi([z]_1^{|s|}, [t^\psi]_1^{|s|})} \tag{3.2}$$

where

$$\Psi([z]_1^{|s|}, [t]_1^{|s|}) = \sum_{1 \leq i \leq |s|} (T_{t_{i-1}, t_i} + z_{t_i}) \tag{3.3}$$

Here $Q^{|s|}$ is a set containing all possible tag sequences of length $|s|$, t_j is the tag for the j^{th} word. The most probable tag sequence is estimated using the Viterbi algorithm [Rabiner, 1989, Collobert et al., 2011].

3.3.4 Training and Implementation

We use a cross-entropy loss function to train the model. Adam's technique [Kingma and Ba, 2014] is used to obtain the optimized values of model parameters. We use a mini batch size of 50 in training for all tasks. In all experiments, we use pre-trained word embeddings of 100 dimensions, which were trained on PubMed corpus using GloVe [Pennington et al., 2014, TH et al., 2015], PoS tag embedding vectors of 10 dimensions, character-based word embeddings of length 20 and a hidden layer size of 250. We use l_2 regularization with 0.001 as the corresponding parameter value. These hyperparameters are obtained using the validation set of the *disease NER* task. We considered batch sizes of $25, 50, 75$ and 100, hidden layer sizes of $150, 200, 250$ and 300 and l_2 regularization with values of $0.1, 0.01, 0.001$ and 0.0001 in tuning the hyperparameters in a greed search. The corresponding training, validation, and test sets for the *disease NER* task are available as separate files with the NCBI disease corpus. For the other two tasks, we used the same set of hyperparameters as those obtained for the *disease NER* task. The same strategy is used for tuning hyperparameters of baseline methods. The entire implementation is carried out in the Python language using the *TensorFlow*[2] library.

3.4 The Benchmark Tasks

In this section, we briefly describe the three standard tasks on which we evaluate the performance of the *CWBLSTM* model. Statistics regarding the corresponding benchmark datasets are provided in Table 3.1.

[2]https://www.tensorflow.org

Dataset	*Corpus*	*Train set*	*Test set*
disease NER	sentences	5661	961
	disease	5148	961
drug NER	sentences	6976	665
	drug	9369	347
	brand	1432	59
	group	3381	154
	drug_n	504	120
clinical NER	sentences	8453	14529
	problem	7072	12592
	treatment	2841	9344
	test	4606	9225

Table 3.1: Statistics of benchmark datasets for the three tasks used in the study.

Disease NER

Identifying disease named entities in text is crucial for disease related knowledge extraction [Bundschus et al., 2008, Agarwal and Searls, 2008]. Furthermore, it has been observed that diseases are one of the most widely searched objects by users in PubMed [Doğan and Lu, 2012]. We use the NCBI disease corpus[3] to investigate the performance of the model on the disease NER task. This dataset was annotated by a team of 12 annotators (2 persons per annotation) on 793 PubMed abstracts [Doğan and Lu, 2012, Dogan et al., 2014].

[3] https://www.ncbi.nlm.nih.gov/CBBresearch/Dogan/DISEASE/

Drug NER

Identifying names of drugs or pharmacological substances is an important first step for drug-drug interaction extraction and other drug-related knowledge extraction tasks. With this in mind, a challenge involving the recognition and classification of pharmacological substances in text was organized as part of SemEval 2013. We use SemEval-2013 task 9.1 [Segura-Bedmar et al., 2013] dataset for this task. The dataset created for this challenge was collected from two sources *i.e.* DrugBank[4] documents and MedLine[5] abstracts. This dataset has four kinds of drugs as entities, namely *drug*, *brand*, *group* and *drug_n*. Here, *drug* represents a generic drug name, *brand* is the brand name of a drug, *group* is a family name of drugs and *drug_n* is an active substance not approved for human use [Segura Bedmar et al., 2011]. In this dataset, while parsing the data, 79 entities (56 *drug*, 18 *group* and 5 *brand*) from the training set and 5 entities (4 *drug* and 1 *group*) from the test set were not recognized by the parser. The entities are treated as the false negative in our evaluation scheme.

Clinical NER

The publicly available (under license) I2B2/VA[6] challenge dataset [Uzuner et al., 2011] is used for clinical entity recognition. This dataset is a collection of discharge summaries obtained from Partners Healthcare, Beth Israel Deaconess Medical Center, and the University of Pittsburgh Medical Center. The dataset was annotated with three kinds of entities, namely *problem*, *treatment* and *test*. Here *problems* denotes phrases that contain observations made by patients or clinicians about the patients' body or mind that are thought to be abnormal or caused by a disease.

[4] `https://www.drugbank.ca/`

[5] `https://www.nlm.nih.gov/bsd/pmresources.html`

[6] `https://www.i2b2.org/NLP/Relations/Main.php`

Treatments are phrases that describe procedures, interventions or substances given to a patient to resolve a medical problem. *Tests* are procedures, and measures that are undertaken on a patient or a bodily fluid or sample to discover, rule out, or find more information about a medical problem.

The downloadable dataset for this task is only a part (only discharge summaries from Partners Healthcare and Beth Israel Deaconess Medical Center) of the full dataset originally used in the challenge. We performed our experiments on the currently available partial dataset. The dataset is available in the preprocessed form, in which sentence and word segmentations has already been carried out. We removed patients' information from each discharge summary before training and testing, because this never contains entities of interest.

3.5 Results and Discussion

3.5.1 Experimental Design

We perform separate experiments for each task. We use the *training set* to learn optimal parameters of the model for each dataset, while evaluation is performed on the *test set*. The performance of each trained model is evaluated based on the strict matching criteria, where exact span boundaries as well as the class need to be correctly identified in order to count as a true positive. To apply the strict matching evaluation scheme, we use CoNLL 2003 evaluation script to calculate precision, recall and F1 score for each task [Tjong Kim Sang and De Meulder, 2003b].

3.5.2 Baseline Methods

We use the following methods as common baselines for comparison with the proposed models in all of the considered tasks. We have implemented all of selected baseline methods, and their corresponding hyperparameters are tuned similar strategies to

those used in the proposed methods.

SENNA: SENNA applies a window based neural network to the embedding of a word with its context to learn global features [Collobert et al., 2011]. To make an inference, it also applies a CRF to the output of the window based neural network. We set the window size to five based on hyperparameter tuning using the validation set (20% of the training set), while the remaining hyperparameters have similar values to our model.

CharWNN: This model [dos Santos and Zadrozny, 2014] is similar to SENNA but uses word as well as character-based embedding in the chosen context window [dos Santos and Guimarães, 2015]. Here character-based embeddings are learned through a convolution neural network with max pooling scheme.

CharCNN: This method [Sahu and Anand, 2016] is similar to the proposed model *CWBLSTM* but instead of using a BLSTM, it uses a convolution neural network to learn character-based embedding.

3.5.3 Comparison with Baseline

Table 3.2 presents a comparison of *CWBLSTM* with different baseline methods on disease, drug and clinical entity recognition tasks. We can observe that CWBLSTM outperforms all three baselines on each of the three tasks. In particular, when comparing with *CharCNN*, the differences are significant for drug NER and disease NER tasks but difference is insignificant for clinical NER. The proposed model improved the recall by 5% resulting about 2.5% relative improvement in F score over the second best method, *CharCNN* for the *disease NER* task. For the drug NER task, a relative improvement of more than 3% is observed for all three measures, precision, recall and F score, over the *CharCNN* model. The relatively weaker performance on clinical NER task could be attributed to the use of many non-standard acronyms and abbreviations in clinical texts which makes it difficult for

Tasks	Models	Accuracy	Precision	Recall	F Score
Disease NER	*SENNA*	97.26	77.93	76.80	77.36
	CharWNN	97.24	78.34	78.67	78.50
	CharCNN	97.61	84.26	78.56	81.31
	CWBLSTM	**97.77**	**84.42**	**82.31**	**83.35**
Drug NER	*SENNA*	96.71	66.93	62.70	64.75
	CharWNN	97.07	69.16	69.16	69.16
	CharCNN	97.09	70.34	72.10	71.21
	CWBLSTM	**97.46**	**72.57**	**74.60**	**73.57**
Clinical NER	*SENNA*	91.56	80.30	78.85	79.56
	CharWNN	91.42	79.96	78.12	79.03
	CharCNN	93.02	83.65	83.25	83.45
	CWBLSTM	**93.19**	**84.17**	**83.20**	**83.68**

Table 3.2: Performance comparison of the proposed model *CWBLSTM* with baseline models on the test sets of different datasets. Here *Accuracy* represents token level accuracy in tagging.

character-based embedding models to learn appropriate representations.

We perform an approximate randomization test [Hoeffding, 1952, Koehn, 2004] to determine if the observed differences in performance between the proposed model and the baseline methods are statistically significant. We considered $R = 2000$ in the approximate randomization test. Table 3.3 shows the p-values of the statistical tests. As the p-values indicate, *CWBLSTM* significantly outperforms *CharWNN* and *SENNA* in all three tasks (significance level: 0.05). However, *CWBLSTM* can outperform *CharCNN* only on the disease NER task.

One can also observe that, even though drug NER has a sufficiently large training dataset, all models achieved a relatively poor performance compared to the

Model	CharCNN	CharWNN	SENNA
Disease NER	0.043	0.49×10^{-3}	0.48×10^{-3}
Drug NER	0.090	0.21×10^{-2}	0.49×10^{-3}
Clinical NER	0.088	0.49×10^{-3}	0.21×10^{-3}

Table 3.3: p-values of the statistical significance test in comparing the performance of the **CWBLSTM model** with other models.

other two tasks. One reason for this poor performance could be the nature of the dataset. As previously discussed the drug NER dataset comprises of texts from two sources, DrugBank and MedLine. Sentences from DrugBank which are written by medical practitioners, are shorter and less comprehensive than MedLine sentences, which are from research articles and generally tend to be longer. Furthermore, the *training set* comprises 5675 sentences from DrugBank and 1301 from MedLine, whereas this distribution is reversed in the *test set*. The test set contains 520 sentences from MedLine and only 145 sentences from DrugBank. The smaller set of training instances from MedLine sentences do not provides sufficient examples for the model to learn.

3.5.4 Comparison with Other Methods

In this section, we compare our results with other existing methods reported in literature. Here, we consider results which are presented in the respective articles. We do not compare our results with the others for the clinical NER task, as the complete dataset (as was available in the I2B2 challenge) is not available and results in literature are reported with respect to the complete dataset.

Model	Features	Precision	Recall	F Score
CWBLSTM	Word, PoS and Character Embedding	84.42	**82.31**	**83.35**
BANNER [Doğan and Lu, 2012]	Orthographic, morphological, syntactic	-	-	81.8
BLSTM+We [Sahu and Anand, 2016]	Word Embedding	**84.87**	74.11	79.13

Table 3.4: Performance comparison of *CWBLSTM* with other existing models on the *disease NER* task

Disease NER

Table 3.4 compares the performance of *CWBLSTM* on the NCBI disease corpus with other existing methods. *CWBLSTM* improves upon the performance of BANNER by 1.89% F Score. BANNER is a CRF-based method which primarily uses orthographic, morphological and shallow syntactic features [Leaman and Gonzalez, 2008]. Many of these features are specially designed for biomedical entity recognition tasks. The proposed model also gave better performance than another BLSTM based model [Sahu and Anand, 2016] by improving recall by around 12%. The BLSTM+WE model [Sahu and Anand, 2016] used a BLSTM network with word embeddings only, whereas the proposed model makes use of extra features in terms of PoS as well as character-based word embeddings.

Drug NER

Table 3.5 compare the performance of *CWBLSTM* on the drug NER task with the submitted results of the SemEval-2013 Drug Name Recognition Challenge [Segura-Bedmar et al., 2013]. *CWBLSTM* outperforms the best result obtained in the challenge (WBI-NER [Rocktäschel et al., 2013]) by a margin of 1.8% in terms of F score. WBI-NER is an extension of the ChemSpot chemical NER [Rocktäschel et al., 2012] system, which is a hybrid method for chemical entity recognition. ChemSpot primarily uses dictionary features to create a sequence classifier using CRF. Additionally, WBI-NER includes features obtained from different domain-

Model	Features	Precision	Recall	F Score
CWBLSTM	Word,PoS and Character Embedding	72.57	**74.05**	**73.30**
WBI [Rocktäschel et al., 2013]	ChemSpot and Ontologies	73.40	69.80	71.5
LASIGE [Grego et al., 2013]	Ontology and Morphological	69.60	62.10	65.6
UTurku [Björne et al., 2013]	Syntactic and Contextual	**73.70**	57.90	64.8

Table 3.5: Performance comparison of *CWBLSTM* with other existing models submitted in the SemEval-2013 *drug NER* task

dependent ontologies. The performance of our proposed model is better than the LASIGE [Grego et al., 2013] and UTurku [Björne et al., 2013] systems by a significant margin. LASIGE is also a CRF based method, and UTurku uses the *Turku Event Extraction System* (TEES) which is a kernel based model for entity and relation extraction tasks.

3.5.5 Feature Ablation Study

We analyze the importance of each feature type by performing feature ablation. The corresponding results are presented in Table 3.6. In this table, the first row represents the performance of the proposed model using all feature types in all three tasks and the second, third and fourth rows show performance when character-based word embeddings, PoS tag embeddings, and pre-trained word embeddings are removed from the model. Removal of pre-trained word embeddings implies the use of random vectors in place of pre-trained vectors.

From the table, we can observe that the removal of character-based word embeddings leads to 3.6%, 5.8% and 1.1% of relative decrement in the F Score on the disease NER, drug NER, and clinical NER tasks respectively. This demonstrates the importance of character-based embedding. As mentioned earlier character-based word embedding helps our model in two ways: (1) it provides morphologically-rich vector representation and (2) vector representations for out-of-vocabulary (OoV)

Model	disease NER	drug NER	clinical NER
CWBLSTM	(84.42), (82.31), (83.35)	(72.57), (74.60), (73.57)	(84.17), (83.20), (83.68)
- CE	(80.86), (80.02), (80.44)	(64.29), (75.62), (69.50)	(83.76), (81.74), (82.74)
- (CE+PE)	(82.72), (77.73), (80.15)	(65.96), (73.42), (69.49)	(83.31), (80.51), (81.89)
- (CE+PE+WE)	(79.66), (73.78), (76.61)	(65.40), (55.80), (60.22)	(79.53), (78.28), (78.90)

Table 3.6: Performance of the model and its variants in a feature ablation study. In every block (X), (Y), (Z) denotes precision, recall and F score, respectively. Here removal of *WE* represents use of random vectors in place of pre-trained word vectors for the word embedding matrix.

Dataset	Unique Words	OoV	Percent
disease NER	8270	819	9.90
drug NER	9447	1309	13.85
clinical NER	13000	2617	20.13

Table 3.7: Statistics of number of words not found in the word embedding files for the different datasets. Here OoV indicates the number of words not found in the pre-trained word embeddings and *percent* indicates the percentage of OoV words in the complete vocabularies of each task.

words can be obtained from character-based word embeddings. OoV words represent 9.9%, 13.85% and 20.13% of the unique words in the drug NER, disease NER and clinical NER dataset respectively (shown in Table 3.7). As discussed earlier, the lower performance of model in clinical NER is because of the high-frequency presence of acronyms and abbreviations, which means that the model cannot take advantage of character-based word embeddings. From the third row of the table, we can also observe that using PoS tag embeddings as a feature is not critical in any of the three tasks. This is because distributed word embedding implicitly preserves that kind of information.

Main Word	Nearest Neighbor Words
cough	coughing, breathlessness, dyspnea, wheezing, coughs
tumor	tumor, tumoral, tumoural, melanoma, tumors
surgery	operation, decompression, dissection, resection, parathyroidectomy

Table 3.8: Lists of words and their five nearest neighbors obtained through pre-trained word embeddings used in the model

In contrast to PoS tag embeddings, we observe that use of pre-trained word embeddings is one of the most important feature types in our model for each task. Pre-trained word embeddings improve the model performance by utilizing a large unlabeled corpus. This helps the model to recognize entities in two ways: Firstly, in the training dataset, there are words which appear very few times (in our case approximately 10% of words in all datasets appear only once in the dataset). During training, their randomly initialized vectors would not be sufficiently updated. On the other hand, as mentioned earlier, pre-trained word embeddings are obtained through training a model on a large unlabeled corpus. Thus, by using the pre-trained word embeddings, we can obtain appropriate vectors for the words which are rare in the training dataset. Secondly, word embeddings preserve lexical and semantic properties in their embedding, which implies that words with similar meanings are assigned similar vectors [Bengio et al., 2003] and that similar words would have a similar label for entity recognition. We can observe from Table 3.8 that most of the words in the nearest neighbor set would also have the same label as the main words.

3.5.6 Effects of CRF and BLSTM

We also analyze the unified framework to gain insights into the effects of using different loss functions in the output layer (CRF vs. WLL) as well as effects of using bi-directional or uni-directional (forward) LSTMs. For this analysis, we modify our

Model	disease NER	drug NER	clinical NER
CWBLSTM	(84.42), (82.31), (83.35)	(72.57), (74.60), (73.57)	(84.17), (83.20), (83.68)
BLSTM+WLL	(76.04), (78.25), (77.13)	(71.81), (70.34), (71.07)	(77.35), (80.91), (79.09)
LSTM+WLL	(64.72), (77.32), (70.46)	(68.41), (69.02), (68.71)	(58.32), (68.11), (62.83)

Table 3.9: Effects of using CRF, WLL and BLSTM in the proposed model on different datasets. In every block (X), (Y), (Z) denote precision, recall and F1 score, respectively.

framework and named model variants as follows: the bi-directional LSTM with WLL output layer is called *BLSTM+WLL* and the uni-directional or regular LSTM with a WLL layer is called *LSTM+WLL*. In other words the *BLSTM+WLL* model uses all the features of the proposed framework, except that it uses WLL in place of CRF. Similarly, *LSTM+WLL* also uses all features, along with a forward LSTM instead of bi-directional LSTM and WLL in place of CRF. The results are presented in Table 3.9. The relative decrements of 7.5%, 5.5% and 3.4% in F Score on disease NER, clinical NER and drug NER tasks, respectively obtained, by *BLSTM+WLL* compared to the proposed model demonstrate the importance of using a CRF layer. This suggests that identifying labels independently is not favored by the model and it is better to use the implicit label dependencies. Calculation of the average token length of entities in three tasks indicates a plausible reason for the difference in performance for the three tasks. The average token length is 2.2 for disease entities, 2.1 for clinical entities and 1.2 for drug entities. The longer the average length of entities, the better the performance of the model utilizing tag dependencies. Similarly, relative improvements of 12.89%, 4.86% and 20.83% in F score on the disease NER, drug NER and clinical NER tasks respectively are observed when *CWBLSTM* is compared with *LSTM+WLL*. This clearly indicates that the use of a bi-directional LSTM is always advantageous.

Word	Char BLSTM	GloVe
2C19	2C9, 2C8/9, 29, 28.9, 2.9z	NA
synergistic	septic, symptomatic, synaptic, serotonergic, synthetic	synergism, synergy, antagonistic, dose-dependent, exerts
dysfunction	dysregulation, desensitization, dissolution, addition, admistration	impairment, impaired, disturbances, deterioration, insufficiency
false-positive	false-negative, facultative, five, folate, facilitate	false, falsely, erroneous, detecting, unreliable
micrograms/mL	microg/mL micromol/L micrograms/ml mg/mL mimicked	NA

Table 3.10: Words and their five nearest neighbors (from left to right in increasing order of Euclidean distance) learned by character-level word embeddings of our model on drug NER corpus. NA denotes that the word is not present in the vocabulary list of the GloVe vectors

3.5.7 Analysis of Learned Word Embeddings

Next, we analyze the characteristics of the learned word embeddings after training of the proposed model. As mentioned earlier, we learn two different representations of each word, one using its characters and the other using it distributional contexts. We expect that the word embeddings obtained through character embeddings will focus on morphological aspects, whereas distributional word embeddings focus on semantic and syntactic contexts.

We obtained, character-based word embedding for each word of the *drug NER* dataset after training. We picked five words from the vocabulary list of the test set and observe their five nearest neighbors in the vocabulary list of the training set. The nearest neighbors are selected using both types of word embeddings, and the results are shown in Table 3.10. We can observe that the character-based word embeddings primarily focus on morphologically similar words, whereas distributional word embeddings preserve semantic properties. This clearly suggests that it is

important to use the complementary nature of the two types of embeddings.

3.6 Conclusion

In this chapter, we have presented a unified model for drug, disease and clinical entity recognition tasks. Our model, called *CWBLSTM*, uses BLSTMs in a hierarchy to learn better feature representations and CRF to infer the correct labels for each word in the sentence simultaneously. *CWBLSTM* outperforms all the baselines in all three tasks. By carrying out various analyses, we demonstrated the importance of each feature type used by *CWBLSTM*. Our analyses suggest that pre-trained word embeddings and character-based word embeddings play complementary roles, and along with the incorporation of tag dependencies, are essential ingredients for improving the performance of NER tasks in the biomedical and clinical domains.

Chapter 4

Convolution Neural Network for Relation Classification

4.1 Overview

In the previous chapter, we examined how different neural networks can be used in a hierarchy to learn morphologically and contextually rich feature representations for different biomedical and clinical entity recognition tasks. This chapter investigates the use of convolution neural network (CNN) in the another subtask of information extraction, namely, the task of relation classification. Here, we assume that all the entities of interest in the texts are already given and, we need to classify the semantic relation between the entities into one of a set of predefined categories. Our model uses CNN with a max pooling scheme to learn an optimal feature representation over word level embeddings for the sentence. We use a softmax classifier on the output of the pooling layer to predict the correct class of relation, or no relation existence. The entire model is trained in an end-to-end manner with adam optimization technique. We refer to this model as *CNN-RE* model in our subsequent discussions.

4.2 Introduction

In recent years, extracting relevant information from biomedical and clinical texts such as research articles, discharge summaries, or electronic health records has been the subject of many research efforts and shared challenges. The automatic extraction of relevant information from these resources can be useful for many applications, such as drug repositioning, medical knowledge base creation, etc. The performance of concept entity recognition systems for detecting the mention of proteins, genes, drugs, diseases, tests, and treatments have achieved a sufficient level of accuracy. This in turn gives us an opportunity to use these data for relation classification, another sub-task of information extraction. Relation classification is the process of identifying how given entities are semantically related in a sentence. As shown in the example sentence [S1] below, the entities *Lasix* and *congestive heart failure* are related by the *treatment administered for medical problem* relation. Such relations are important for further upper-level NLP tasks and also for biomedical and clinical research [Shang et al., 2011].

[S1]: *He was given* **Lasix** *to prevent him from* **congestive heart failure**.

In the literature, relation classification tasks in biomedical text have been modeled in several ways. *Co-occurrence* based methods, due to their simplicity and flexibility, are the most widely used. In co-occurrence methods, it is assumed that if two entities occur together in many sentences, then there must be a relation between them [Bunescu et al., 2006, Song et al., 2011]. However, this method suffers from low precision and low recall as the assumption does not take into account contextual information. Furthermore, this method is not designed to identify the exact semantic relation type. Rule-based methods are another commonly used method for relation classification task [Thomas et al., 2000, Park et al., 2001, Leroy et al., 2003]. Rules are created by carefully observing the syntactic and semantic patterns of relation instances or through domain knowledge.

Feature-based methods use feature extraction techniques to utilize contextual information present in a sentence containing predefined entities to extract a vector representation [Zelenko et al., 2003, Culotta and Sorensen, 2004, Hong, 2005, Minard et al., 2011, Qian and Zhou, 2012, Zeng et al., 2014]. Feature extraction is mainly based on the output of linguistic and domain-specific tools. The extracted feature vectors are then used by a classification technique to decide the correct class of relation present between entities in the sentence. State-of-the art results have been obtained by this class of methods. However, the performance of feature-based methods is highly dependent on the selection of a suitable feature, which is not only a tedious and time-consuming task, but also requires domain specific tools. Another problem faced by these methods is that feature extraction needs to be adjusted according to the data source. As discussed earlier, our dataset consist of multiple but diverse information resources such as research articles, discharge summaries, clinical notes, etc. While on one hand, multiple sources bring more information, on the other hand, they make it challenging to extract meaningful information automatically, simply because of diverse characteristics of the data sources. For example, sentences in research articles are well formed, and likely to use only well-accepted technical terms. In contrast, sentences in clinical discharge summaries may not be well-formed, instead, they will often be fragmented sentences with many acronyms or terms used only locally. Similarly, social media articles may use slang or terms which are not technical. This makes it difficult to design a single set of features that is suitable for all of the above text types.

Motivated by these issues, this work aims to exploit recent advances in the machine learning and NLP domains to reduce such dependencies and utilizes CNN to learn important features with minimal manual dependencies. CNN has been shown to be a dominant model to solve problems in image processing and computer vision [Krizhevsky et al., 2012,Karpathy and Fei-Fei, 2014]. Subsequently, in natural

language processing, it has also demonstrated state-of-the-art results in different tasks, such as sentence classification [Kim, 2014, Kalchbrenner et al., 2014, Hu et al., 2014, Sharma et al., 2016], relation classification [Zeng et al., 2014, dos Santos and Guimarães, 2015] and semantic role labeling [Collobert et al., 2011]. In this chapter, we investigate the use of CNN to extract relations in biomedical and clinical texts. In particular, we use dataset developed for clinical relation classification (CRC) task organized by Informatics for Integrating Biology and the Bedside (I2B2) in 2010, as part of I2B2/VA challenge [Uzuner et al., 2011], and the drug-drug interaction extraction (DDI) task, organized at SemEval 2013 [Segura-Bedmar et al., 2013]. The CRC dataset consists of discharge summaries and progress report of patients. Three types of entities, *i.e. problem*, *treatment* and *test*, are present in the dataset. In contrast, the DDI dataset consists of MedLine abstracts and DrugBank documents, and has different types of drug names as entities.

4.3 Model Architecture

The proposed model architecture is shown in Figure 4.1. It takes a complete sentence with mentioned entities as input and outputs a probability vector corresponding to all possible relation types. The first layer of the model is a feature layer which uses exact word and distance from the first and the second targeted entities as features. Each feature has a vector representation which is initialized randomly, except for the word embedding feature. For the word embedding feature, we use pre-trained word vectors learned from PubMed articles using the *GloVe* embedding tool. The embedding layer maps every feature value with its corresponding feature vectors and concatenates them. In order to obtain local features from each part of the sentence, we have used multiple filters of different lengths [Kim, 2014] in all possible continuous n-grams of the sentence, where n is the length of the filter. We use max pooling over time to obtain global features through all filters. Here, time indicates

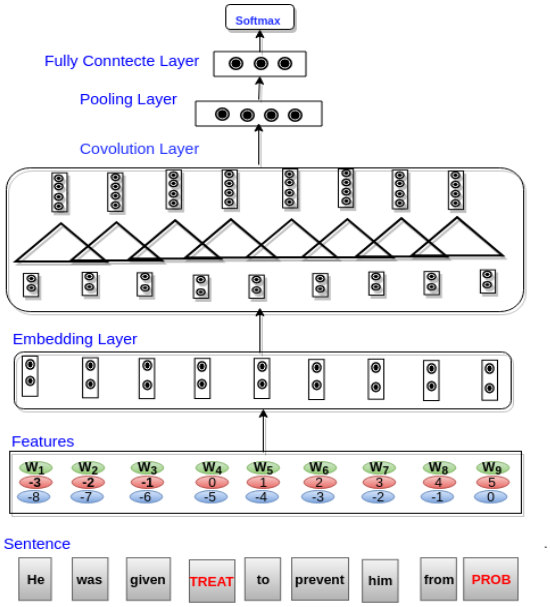


Figure 4.1: CNN-RE model for relation classification.

that the filter is running over the length of the sentence. Pooled features are then fed to a fully connected feed-forward neural network to make the inference. In the output layer, we use a softmax classifier whose number of nodes is equal to the number of possible relation types between entities. We next describe each part of the model in detail.

Feature Layer

We represent each word in a given sentence with three discrete features, namely, the word itself (W), the distance from the first entity (P_1) and the distance from the second entity (P_2). Each feature is briefly described below:

1. W : Exact word as it appeared in the sentence.

2. P_1: The distance from the first entity in terms of the number of words [Collobert and Weston, 2008]. For instance in our earlier example [S1] *He* is at a

distance of -3 and *prevent* is at a distance of $+2$ from the first entity *Lasix*. This value would be zero for all words which are a part of the first entity.

3. P_2: Similar to P_1 but considers distance from the second entity.

In this way a word $w \in D^1 \times D^2 \times D^3$, where D^i is the dictionary for i^{th} local feature.

Embedding Layer

In the embedding layer, each feature value is mapped to its vector representation using a feature embedding matrix. Let $M^i \in \mathbb{R}^{n_i \times N_i}$ be the feature embedding matrix for i^{th} feature (here n_i represents the number of dimensions of feature embedding and N_i is the number of possible values or size of the dictionary for i^{th} local feature). Each column of M^i is a vector of corresponding values of the i^{th} feature. Mapping can be done by taking the product of one hot vector of a feature value with its embedding matrix [Collobert and Weston, 2008]. Suppose $a_j^{(i)}$ is the one hot vector for the j^{th} feature value of the i^{th} feature, then:

$$f_j^{(i)} = M^i \, a_j^{(i)} \tag{4.1}$$

$$x^i = f_1^{(i)} \oplus f_2^{(i)} \oplus f_3^{(i)} \tag{4.2}$$

Here $\oplus$ is the concatenation operation so $x^i \in \mathbb{R}^{(n_1+n_2+n_3)}$ is the feature vector for the i^{th} word of sentence and n_k is the dimension of the k^{th} feature. For word embeddings, we used pre-trained word vectors obtained by running the *GloVe* tool on a corpus of PubMed open source articles. The other feature matrices are initialized randomly at the beginning.

Convolution Layer

We apply convolution on text to obtain local features from each part of the sentence [Collobert and Weston, 2008]. Consider $x^1x^2.....x^m$ to be the sequence of feature vectors of a sentence, where $x^i \in \mathbb{R}^d$ is a vector obtained by concatenating all feature vectors of the i^{th} word. Let $x^{i:i+j}$ represent the concatenation of $x^i.....x^{i+j}$ feature vectors. Suppose there is a *filter* parameterized by weight vector $w \in \mathbb{R}^{c.d}$ where c is the length of the filter (in Figure 4.1, the filter length is two). Then, the output sequence of the convolution layer would be

$$h^i = f(w \cdot x^{i:i+c-1} + b) \tag{4.3}$$

where $i = 1, 2, \ldots m - c + 1$, . indicates the dot product, f is a rectified linear unit (ReLu) function and $b \in \mathbb{R}$ is a biased term. w and b are the learning parameters and will remain same for all $i = 1, 2, \ldots m - c + 1$.

Max Pooling Layer

The length $(m - c + 1)$ of the output of convolution layer will vary based on the number of words m in the sentence. We applied max pooling [Collobert and Weston, 2008] over time to obtain fixed length global features for a sentence. The intuition behind using max pooling is to consider only the most useful features from the entire sentence.

$$z = \max_{1 \leq i \leq (m-c+1)} [h^i] \tag{4.4}$$

We have explained the process of extracting one feature from a whole sentence using one filter. In Figure 4.1 we extracted four features using four filters of the same length, *i.e.* two. In our experiments, we use multiple filters of variable length [Kim, 2014]. The objective of using different lengths of filter is to accommodate contexts with varying window sizes around the words.

Fully Connected Layer

The output of the max pooling layer is a vector z. We call this a global feature because it is obtained by taking max over the entire sentence. To create a classifier over the extracted global feature, we use a fully connected feed forward layer. Suppose $z^i \in \mathbb{R}^l$ is the output of the max pooling layer (l represents the number of filters used) then the output of fully-connected layer would be

$$o^{(i)} = W^o z^i + b^o \tag{4.5}$$

Here $W^o \in \mathbb{R}^{r \times l}$ and $b^o \in \mathbb{R}^r$ are parameters of the neural network and r denotes the number of classes.

Softmax Layer

A softmax classifier is used in in the output layer, where we minimize the following objective function:

$$L_i = -\log\left(\frac{e^{o^{(i)}_{y_i}}}{\sum_{\forall j} e^{o^{(i)}_j}}\right) \tag{4.6}$$

where y_i is the correct class of the relation for the i^{th} instance.

4.4 Implementation

We experiment with filter lengths in two different experimental settings. In the first, we use h filters of a fixed length in the convolutional layer, while in the second set of experiments we use k varying length filters. For each of the k varying lengths, h different filters are used. So, in the first setting, we obtain h features after max pooling, while in the second, $h \times k$ features are obtained. For regularization, we use *dropout* [Srivastava et al., 2014] and the l_2 regularization technique in the output of max pooling layer. We use adam technique [Kingma and Ba, 2014] to optimize the

Dataset	*Relations*	*Training Set*	*Test Set*
DDI	Pairs	16495	4025
	Positive	3844	979
	Negatives	12451	3046
DDIC	Pairs	16495	4025
	Int	140	96
	Advice	820	221
	Mechanism	1264	302
	Effect	1620	360
	Negatives	12451	3046
CRC	Pairs	49853	12461
	TeCP	408	102
	TrCP	434	109
	PIP	1775	444
	TrAP	2107	527
	TeRP	2453	614
	Negatives	42658	10665

Table 4.1: Statistics of the datasets used for relation classification tasks.

loss function. The entire set of neural network parameters, as well as the feature vectors, are updated while training. We implemented the proposed model in the Python language using the *tensorflow*[1] package.

4.5 Tasks and Datasets

In this section, we briefly describe the tasks and the corresponding datasets used in this study. Statistics of these datasets are given in Table 4.1.

Task: DDI Extraction

Two or more drugs can affect the activities of each other when administered together [Segura-Bedmar et al., 2013]. Often, such interaction among drugs can have severe

[1] `https://www.tensorflow.org/`

adverse consequences [Businaro, 2013]. Identifying DDIs present in a text is a kind of relation classification task. In this task, given two drugs or pharmacological substances in a sentence, we need to classify their interaction into one of the four interaction categories (discussed later) or determine that no interaction exist between them.

Ex.1: *Lithium*$_{drug}$ *generally should not be given with diuretics*$_{drug}$

In *Ex.1*, the drugs *Lithium* and *Diuretics* interact with each other and type of interaction is *advice* because the sentence suggests that they should not be given together. For the DDI Extraction task, we use the dataset from SemEval 2013[2] DDI Extraction challenge [Herrero-Zazo et al., 2013, Segura-Bedmar et al., 2013]. This dataset contains annotated sentences from two sources, *i.e.*, MedLine abstracts and the DrugBank database. MedLine contains biomedical research articles and DrugBank contains manually curated texts collected from various sources and verified by accredited experts. The dataset was annotated with the following four kinds of interactions:

Advice: The text states an opinion or recommendation related to the simultaneous use of the two drugs, *e.g.*, "*alpha-blockers* should not be combined with *uroxatral*".

Effect : The sentence notes the effect of the drug-drug interaction or the pharmacodynamic mechanism of the interaction. For example "*Warfarin* users who initiated *fluoxetine* had an increased risk of hospitalization for gastrointestinal bleeding".

Mechanism : The sentence describes a pharmacokinetic mechanism, as in "*Paroxetine* reduces the plasma concentration of *endoxifen* by about 20%".

Int : The text mentions a drug interaction without providing any other information. For example, "This is typical of the interaction of *meperidine* and

[2] `https://www.cs.york.ac.uk/semeval-2013/task9/`

MAOIs.".

We use the dataset in two separate tasks, namely, DDI detection which we call the **DDI task** and DDI classification, which we refer to as the **DDIC task**. The DDI task corresponds to the detection of a drug interaction in a given sentence for a given pair of drugs. For this, we ignore the specific interaction type and class, and instead, consider that all types of interactions belong to a single positive class, while instance where there is no interaction instances are kept in the negative class. However, in the DDIC task, we have to determine either the exact class of interaction (one of the four types introduced above) or that no interaction exists.

Task: Clinical Relation Classification (CRC)

Clinical relation classification is the task of identifying relations among clinical entities such as *Problem, Treatment* and *Test* in clinical notes or discharge summaries. For instance, in *Ex.2*, *allergic* and *rash* have *medical problem indicates medical problem* relation.

Ex.2: *She is allergic*$_{Problem}$ *to augmentin which gives her a rash*$_{Problem}$.

I2B2 released a dataset for clinical relation classification as a part of *I2B2/VA-2010* shared challenge [Uzuner et al., 2011]. This dataset consist of documents collected from three different hospitals, and was manually annotated by medical practitioners with problem, treatment and test entities, and eight relation types among these entities. The relations are:

Treatment caused medical problems (**TrCP**)

Treatment administered for medical problem (**TrAP**)

Treatment worsens medical problem (**TrWP**)

Treatment improves or cure medical problem (**TrIP**)

Treatment was not administered because of medical problem (**TrNAP**)

Test reveals medical problem (**TeRP**)

Test conducted to investigate medical problem (**TeCP**)

Medical problem indicates medical problems (**PIP**).

The original challenge dataset consist of 394 documents for training and 477 documents for testing. However, only 170 documents for training and 256 documents for testing were available for download (under the license agreement) when we accessed the website. We decided to remove instances of the *TrWP* and *TrIP* classes as the partial dataset does not have sufficient instances of these in its training and test sets. The statistics of the dataset are presented in Table 4.1.

Preprocessing

As a preprocessing step, we replace the texts of the entities in the I2B2 dataset with the corresponding entity type labels. For instance, the sentence: "He was given *Lasix* to prevent him from *congestive heart failure*" was converted to: "He was given *TREAT* to prevent him from *PROB*". Here TREAT stands for *Treatment* and PROB stands for *Problem*. Similarly, for the DDI Extraction dataset, the two targeted drug names are replaced with the labels DRUG-A and DRUG-B respectively. If there are other drug names present in the same sentence, then they are replaced with DRUG-N. Furthermore, all numbers were replaced with the label *DG*. Similarly to previous studies [Rastegar-Mojarad et al., 2013, Kim et al., 2015, Liu et al., 2016b], a small number of few negative instances is filtered from the DDI Extraction dataset by applying rules. These rules have not eliminated any positive instances from the test set. However, 144 positive instances (54 *Mechanism*, 65 *Effects*, 49 *Int* and 6 *Advice*) are removed from the training set by the rules. The statistics shown in Table 4.1 shows the number of instances after negative instance filtering rules to the complete dataset. We filter negative samples based on the following rules:

1. If both the targeted drug mentions have the same name, remove the

corresponding instance. The assumption behind this rule is that a drug cannot interact with itself. We use string matching on both drug names to identify such cases.

2. Remove the instance, if one drug is a kind of or a special case of the other drugs in the corresponding sentence. To identify such cases, we use regular expressions to match patterns in the dataset. "*DRUG-A (DRUG-B)*", "*DRUG-A such as DRUG-B*" are examples of such patterns.

3. If both target drugs appear in same coordinate structure, then remove the corresponding instance. We use several regular expressions based on observation of the patterns in training set to filter out such instances. A examples of one such regular expression pattern is "*DRUG-A , (DRUG-N ,)*$^{+}$*DRUG-B*".

4.6 Experiment Design

4.6.1 Hyper-parameters

As there is no separate development or validation set available, we divided the original training datasets of each task into two parts, 80% for training and the remaining 20% for validation. The hyper-parameters are tuned using this validation set. In all our experiments, a hidden layer size of 200 is used for each filter, while pre-trained word embeddings of 100 dimensions and distance embeddings of 10 dimensions are used in all three tasks. Word embeddings are obtained using the *GloVe* tool [Pennington et al., 2014] applied to a corpus of PubMed open source articles [TH et al., 2015]. The different values of the regularization parameters used for each task are shown in Table 4.2.

Tasks	***Dropout***	l_2 ***regu.***
DDI	0.6	0.1
DDIC	0.7	0.1
CRC	0.7	0.01

Table 4.2: Values of the different regularization parameters used in the three tasks.

4.6.2 Baseline Methods for comparison

We compare the performance of the proposed model with several baseline methods. Approaches based on conventional features and kernel methods are included as baseline methods. These methods can be classified into two categories: one-stage and two-stage methods. In one-stage methods, a multi-class classifier is used to map a sentence with two target entities either into one of the relation classes or into the negative class. In contrast, two-stage methods, as the name suggests, break the problem into two steps. The first step builds a binary classifier to determine whether or not a relation exists between two target entities. Only those sentences with target entity pairs, falling into the positive category of the binary classifier in the first step, are considered as input to the multi-class classifier of the second step. Below we briefly describe all the baseline methods, where superscript $1s$ indicates one stage and superscript $2s$ indicates two stages methods.

Linear Methods

In this class of methods, a linear classifier is used to identify the correct class of relation for each instance. All instances are represented using a vector of manually designed features. **UTurku**^{1s} used the Turku event extraction system (TEES) [Björne et al., 2013] for drug-drug interaction extraction. The major features used by TEES comes from dependency parsing and domain-dependent resources such, as

MetaMap[3]. **UWM-TRIADS**^{2s} [Rastegar-Mojarad et al., 2013] and **Kim**^{2s} [Kim et al., 2015] are two stage methods. Both of these methods use SVM with contextual, lexical, semantic and tree structure features in both of the stages.

Kernel Methods

Kernel methods are powerful techniques for utilizing graph-based features in any NLP task. **WBI-DDI**^{2s} and **FBK irst**^{2s} are two stage methods [Chowdhury and Lavelli, 2013a, Thomas et al., 2013] for DDI classification. The first stages of both models emply different kernel methods thus use syntax tree and dependency tree features. In stage two, WBI-DDI^{2s} uses TEES and FBK irst^{2s} uses SVM with a non-linear kernel for classification. **NIL UCM**^{1s} uses a multi-class SVM with kernel methods in the one stage framework.

As discussed earlier, the dataset used in this work for the CRC task is a partial dataset. Therefore, a direct comparison of our model's performance with the results obtained in the I2B2/VA-2010 challenge is not possible. Hence, we created a SVM classifier whose results are considered as a baseline for the CRC task. We call this method **SVM-RE** model. The SVM-RE model uses similar set of features to [Rink et al., 2011].

4.7 Results and Discussion

4.7.1 Influence of Filter Lengths

We train and evaluate the *CNN-RE* model on all three DDI, DDIC and CRC tasks separately. In each case, we use the training set to train a *CNN-RE* model and evaluation is performed using independent test set of each respective task. Firstly, we show the influence of different filter lengths used in the *CNN-RE* model. Table

[3] `https://metamap.nlm.nih.gov/`

Filter length	DDI			DDIC			CRC		
	Prec.	Recall	F Score	Prec.	Recall	F Score	Prec.	Recall	F Score
[3]	75.05	70.99	72.96	70.51	60.57	65.16	77.00	71.77	74.29
[4]	76.01	70.88	73.36	63.47	63.73	63.60	75.06	75.77	75.42
[5]	77.35	70.48	73.75	64.69	63.84	64.26	78.75	72.43	75.46
[6]	76.05	70.07	72.93	71.16	59.75	64.96	75.72	75.38	75.55
[3,4]	73.38	74.05	73.71	67.00	61.38	64.07	75.85	75.38	75.62
[4,5]	78.77	70.88	74.62	72.98	56.28	63.55	77.38	73.71	75.50
[3,4,5]	80.06	70.99	**75.25**	68.70	63.02	**65.74**	77.87	74.27	**76.03**
[3,4,5,6]	82.28	67.82	74.35	66.35	64.86	65.59	72.15	77.89	74.91

Table 4.3: Comparative performance of the CNN-RE model using filters of different lengths separately and together.

4.3 shows the performance of *CNN-RE* model on the three tasks with different filter lengths.

In case of a fixed filter length, we can observe that no single filter length achieves giving the best performance in all the tasks. Specifically, five is the best filter size for the DDI task, three is the best length for the DDIC task and six is the best filter size for the CRC task. The observation that the optimal filter length is task-dependent is to be excepted, because influential words or important features may vary across the different datasets. However, in case of variable filter length, the best performance was obtained by the [3,4,5] filter on all the tasks. We can also observe that, our model obtained better performance by using variable filter lengths in all the tasks. This is an agreement with observations made in other studies [Kim, 2014, Nguyen and Grishman, 2015] using CNN-based models for NLP tasks. As mentioned earlier, influential words or important features may not always lie within the same window length. With a variable length filter, the model has the option for select best among all local features.

Furthermore, one can also observe that the performance on DDI and DDIC tasks are inferior to that on the CRC task, even though DDI and DDIC have

fewer of classes than CRC. The possible reasons could be the heterogeneous data sources and the relatively smaller training data size in the DDI and DDIC tasks. As mentioned in section 4.5, the dataset for both of these tasks comes from two sources, *i.e.*, DrugBank documents and MedLine abstracts. MedLine abstracts are likely to contain longer sentences that use technical terminologies, while DrugBank documents are more concise and specifically concerns drug interaction information. This heterogeneous nature of dataset increases the complexity of relation identification.

4.7.2 Class wise Performance

We took the best combination of filter lengths ([3,4,5]) and examine the class-wise performance of the DDIC and CRC tasks. The results are shown in Table 4.4 and 4.5. We observe that for both the tasks, the *CNN-RE* model has greater difficulty in predicting some relation classes then others. The performance in the CRC task is considerably better for *TeRP*, *TrAP* and *PIP* classes than for the *TeCP* and *TrCP* classes. A possible reason for this is that *TeRP*, *TrAP* and *PIP* are the most common relation instances (88.26% of positive instances) in the dataset of CRC task. Similarly, in the DDIC task, a much worse performance is observed for the *int* class of interaction compared to the *advice*, *mechanism* and *effect* classes. In this case the, *int* class covers only 0.84% of the entire training dataset. Moreover, the best performance in the DDIC task is obtained for the *advice* class, even though it is not the largest interaction class in the training dataset. The reason could be that advice or suggestions regarding drug interactions are typically described using very limited set of similar phrases, *e.g.*, *"should not be used"* or *"caution should be observe"* etc.

Name	Precision	Recall	F Score
Int	86.11	32.29	46.96
Advice	69.19	74.20	71.61
Mechanism	71.70	61.25	66.07
Effect	64.57	65.83	65.19

Table 4.4: Class wise performance of the **DDIC task** (filter size : [3,4,5] each with 100 filters.)

Name	Precision	Recall	F Score
TeCP	79.62	42.15	55.12
TrCP	85.29	26.60	40.55
PIP	72.24	68.01	70.06
TrAP	74.73	80.26	77.40
TeRP	83.77	87.45	85.57

Table 4.5: Class wise performance of the **CRC task** (filter size : [3,4,5] each with 100 filters.)

4.7.3 Feature Ablation Study

In order to investigate the importance of each feature in the final result, we gradually remove different features from *CNN-RE* and retrain the model on the same dataset. Table 4.6 shows the results obtained when only position embedding is removed, and when position embedding is removed as well as using random vectors in place of pre-trained word vectors for the word embedding matrix. 4%, 2.4% and 9.3% of relative decrements in the F scores obtained for the DDI, DDIC and CRC tasks respectively, are observed when the position embedding feature is removed from *CNN-RE* (2^{nd} row). Similarly, 4%, 4.3% and 13.0% of further relative decrements are observed in the performance on the DDI, DDIC and CRC tasks respectively, when neither the position nor pre-trained word embeddings are used (3^{rd} row). This shows the importance of position and word embedding features.

4.7.4 Comparison with Baseline

Table 4.7 shows a detailed comparison of our models with other existing methods on the DDI, DDIC and CRC tasks. We compare our model's performance with the

Filter length	DDI			DDIC			CRC		
	Prec.	Recall	F Score	Prec.	Recall	F Score	Prec.	Recall	F Score
CNN-RE	80.06	70.99	75.25	68.70	63.02	65.74	77.87	74.27	76.03
CNN-RE - {P}	73.98	70.58	72.24	65.22	63.02	64.10	71.91	66.14	68.90
CNN-RE - {(P+X)}	77.14	67.92	72.24	70.18	56.99	62.90	68.79	63.69	66.14

Table 4.6: Performance of the CNN-RE model in a feature ablation study. Here P refers to random position embedding for both P_1 and P_2, and X refers to pre-trained word vector embedding.

top performing methods in the respective challenges. We can observe that our best performing model *CNN-RE* [3,4,5] is competitive with the state-of-the-art in the DDI and DDIC tasks. However, it outperforms the baseline method in the CRC task. All the baseline methods use manually crafted features obtained from other NLP tools. Careful design of features requires considerable time and effort and it is often difficult to reproduce results, because of the lack of explanation of feature engineering in the referred papers that describe the methods.

4.7.5 Error Analysis

We have tried to determine relevant factors which are adversely affecting the performance of the model. For this, we look at the average sentence lengths of instances correctly and incorrectly classified by our models in the DDI and DDIC tasks (Table 4.8). We observe that the average sentence length and the entity separation length (number of words between the targeted entities) for incorrectly classified instances are always higher compared to correctly classified instances. Another feature of many incorrectly predicted instances was the presence of multiple drug entities in the same sentence. Multiple mention of drugs repetitively are more likely to behave like noise, which may cause neural models to disregard relevant information from other words likely to be contextually important. Hence, a better

Tasks	*Models*	*Precision*	*Recall*	*F Score*
DDI	UTurku[1s] [Björne et al., 2013]	**85.8**	58.5	69.6
	Kim[2s] [Kim et al., 2015]	-	-	77.5
	NIL UCM[1s] [Bokharaeian and DIAZ, 2013]	60.8	56.9	58.8
	WBI-DDI[2s] [Thomas et al., 2013]	80.1	72.2	75.9
	FBK irst[2s] [Chowdhury and Lavelli, 2013b]	79.4	**80.6**	**80.0**
	CNN-RE Model	80.06	70.99	75.25
DDIC	UTurku[1s] [Björne et al., 2013]	**73.2**	49.9	59.4
	UWM-TRIADS[2s] [Rastegar-Mojarad et al., 2013]	43.9	50.5	47.0
	Kim[2s] [Kim et al., 2015]	-	-	**67.0**
	NIL UCM[1s] [Bokharaeian and DIAZ, 2013]	53.5	50.1	51.7
	WBI-DDI[2s] [Thomas et al., 2013]	64.2	57.9	60.9
	FBK irst[2s] [Chowdhury and Lavelli, 2013b]	64.6	**65.6**	65.1
	CNN-RE Model	68.70	63.02	65.74
CRC	**SVM-RE**[1s] [Rink et al., 2011]	76.64	72.55	74.54
	CNN-RE Model	**77.87**	**74.27**	**76.03**

Table 4.7: Performance comparison of *CNN-RE* with baseline methods. Performance is measured based on precision, recall and F score. The highest scores are highlighted in bold.

strategy is required to deal with such cases. Considering limited context along with the removal of repeated mentions of entities could be one way to deal with particularly long sentences.

Model	Sentence Length		Entity Separation	
	True	False	True	False
DDI	$28.4_{(16.67)}$	$41.5_{(21.17)}$	$11.77_{(10.47)}$	$14.96_{(11.15)}$
DDIC	$26.39_{(13.68)}$	$42.11_{(22.50)}$	$11.24_{(9.39)}$	$15.17_{(12.21)}$

Table 4.8: Mean and standard deviations (in subscript) of sentence length and entity separation length for True Positive and False Negative instance for the CNN-RE [3,4,5] model.

4.8 Conclusion

In this chapter, we have presented a unified model for biomedical and clinical relation classification tasks. The proposed model, referred to as *CNN-RE*, uses word and position embeddings as features in the input layer and learns an appropriate vector representation for the sentence through CNN and pooling layers. The learned vectors are then used in a softmax classifier to predict either the correct class of the relation or that no relation exists, in the output layer of the *CNN-RE* model. We have evaluated the performance of the proposed model on the DDI, DDIC and CRC tasks. Our results demonstrate that *CNN-RE* can be used as an alternative method to conventional feature-based methods. An analysis of the results reveald the following important points: variable length filters perform better than same length filters; word and position embeddings are essential ingredients for the *CNN-RE* model; imbalance and noise in the dataset adversely affect the performance of the model; and model is more likely to make an incorrect classification for longer sentences.

Chapter 5

LSTM Network for Relation Classification

5.1 Overview

In the last chapter, we investigated the performance of a CNN model for different biomedical and clinical relation classification tasks. We observe that CNN can be used as an alternative to the conventional feature based methods for the relation classification tasks. However, error analysis indicates that the *CNN-RE* model is failing for the sentences of relatively longer length. In such cases, the presence of important clues is lying in discontinuous places. In this chapter, our primary objective is to improve the performance of the relation classification task by using more powerful representation learning ability. Here for learning better feature representation, we investigate the long short-term memory network in place of convolution neural network and attentive pooling scheme in place of max pooling over previous *CNN-RE* model.

5.2 Introduction

Relation classification is the task of identifying the semantic relations present between a given pair of entities in a text. Since most search queries are some forms of binary factoids [Agichtein et al., 2005], modern question-answering systems heavily rely upon relation classification as a preprocessing step [Fleischman et al., 2003, Lee et al., 2007, Girju, 2003]. Accurate relation classification also facilitates discourse processing and precise sentence interpretations. Hence, this task has witnessed lots of attention over the last decade [Mintz et al., 2009, Surdeanu et al., 2012]. In the biomedical domain, in particular, extracting such tuples from data may be essential for identifying protein and drug interactions [Thomas et al., 2000, Qian and Zhou, 2012], symptoms, and causes of diseases [van Mulligen et al., 2012], among others. Further, since clinical data tend to be obtained from multiple (and diverse) information sources such as journal articles, discharge summaries, and electronic patient records, relation classification becomes a more challenging task.

Existing methods can be classified into two categories: those relying on handcrafted features and those using latent features. In the first category, support vector machines (SVMs) with linear or non-linear kernels have mainly been employed in several studies [Chowdhury and Lavelli, 2013b, Bokharaeian and DIAZ, 2013, Kim et al., 2015]. All of these methods depend on manually engineered features such as PoS tags, chunk tags, trigger words, shortest dependency trees and syntax trees. Methods using non-linear kernels map structure features (dependency and syntax trees) to real values. Such methods have successfully been employed for similar relation classification tasks, including ADR extraction [Gurulingappa et al., 2012b, Gurulingappa et al., 2012a, Harpaz et al., 2014], protein-protein interaction extraction texts [Qian and Zhou, 2012], relations between genes and diseases [Bravo et al., 2015], and relations among medical concepts [Rink et al., 2011]. Although such methods have demonstrated effective performance, they require manually crafted

features. However, the extraction of these features is dependent on other NLP tools and the inherent noise and cost of such tools may adversely affect the performance of models that depend on these features. Methods using latent features and belonging to the second category result from re-emergence of deep learning models as a powerful alternative to conventional feature based models. Certain notable studies [Zhao et al., 2016,Sahu et al., 2016] on DDI and CRC tasks are based on convolution neural networks and have been shown to achieve superior performance than the existing state-of-the-art methods.

In this work, we also rely on latent features learned by neural network models. As opposed to works in [Zhao et al., 2016,Sahu et al., 2016], which use CNN models, we use LSTM based neural network models [Hochreiter and Schmidhuber, 1997]. *CNN-RE* model requires pooling on continuous n-grams, constructed on an entire sentence in order to obtain constant length features. Here n is the length of the convolution or filter. This may cause problems for the longer length sentences or those containing relevant clues lying far away from one another. To overcome this issue, we use BLSTM with two different pooling techniques for encoding variable length features. Theoretically, a BLSTM can preserve information regarding the past and future words while reading [Hochreiter and Schmidhuber, 1997]. Therefore, when we apply pooling on the BLSTM output, we can get features containing information on the complete context of the entire sentence. This is in contrast to the *CNN-RE* models which extract features based on the sentence n-gram. With this intuition, we propose three models, namely: *BLSTM-RE*, *ABLSTM-RE* and *Joint ABLSTM-RE* for the CRC and DDI extraction tasks. Here *BLSTM-RE* and *ABLSTM-RE* uses a BLSTM for encoding word and position features. *BLSTM-RE* uses max pooling while *ABLSTM-RE* uses attentive pooling on the BLSTM outputs to obtain fixed length features over the entire sentence. However, as an ensemble of *BLSTM-RE* and *ABLSTM-RE*, *Joint ABLSTM-RE* uses two BLSTMs, one with

max pooling and another with attentive pooling. In each of these models, we use a fully connected neural network with a softmax function in the output layer.

5.3 Model Architecture

We present three BLSTM based models, namely, *BLSTM-RE*, *ABLSTM-RE* and *Joint ABLSTM-RE* for the DDI, DDIC and CRC tasks. A generic architecture of the proposed models is illustrated in figure 5.1. Each model uses embedding features as input in the first layer and learns a fixed length vector representation through subsequent layers. The score for each possible class is computed in the final layer, and the final decision is reached using this score. Training of the model happens in an end-to-end manner such a way that the correct class will get the high score after training. We briefly explain each component of the three models in the following sections.

Feature Layer

We represent each word in the sentence with three discrete features, namely: *word* (W), *distance*$_1$ (P_1), and *distance*$_2$ (P_2). Here W is an exact word appear in the sentence. P_1 indicates the distance (in terms of words) from the first entity name [Collobert et al., 2011, Sahu et al., 2016] and this value would be zero for first targeted entity name word. P_2 is similar to P_1, but denotes the distance from the second targeted entity name. In this manner, a word $w \in D^1 \times D^2 \times D^3$, where D^i is the dictionary for i^{th} local features. This feature layer constitutes the first layer for all the models.

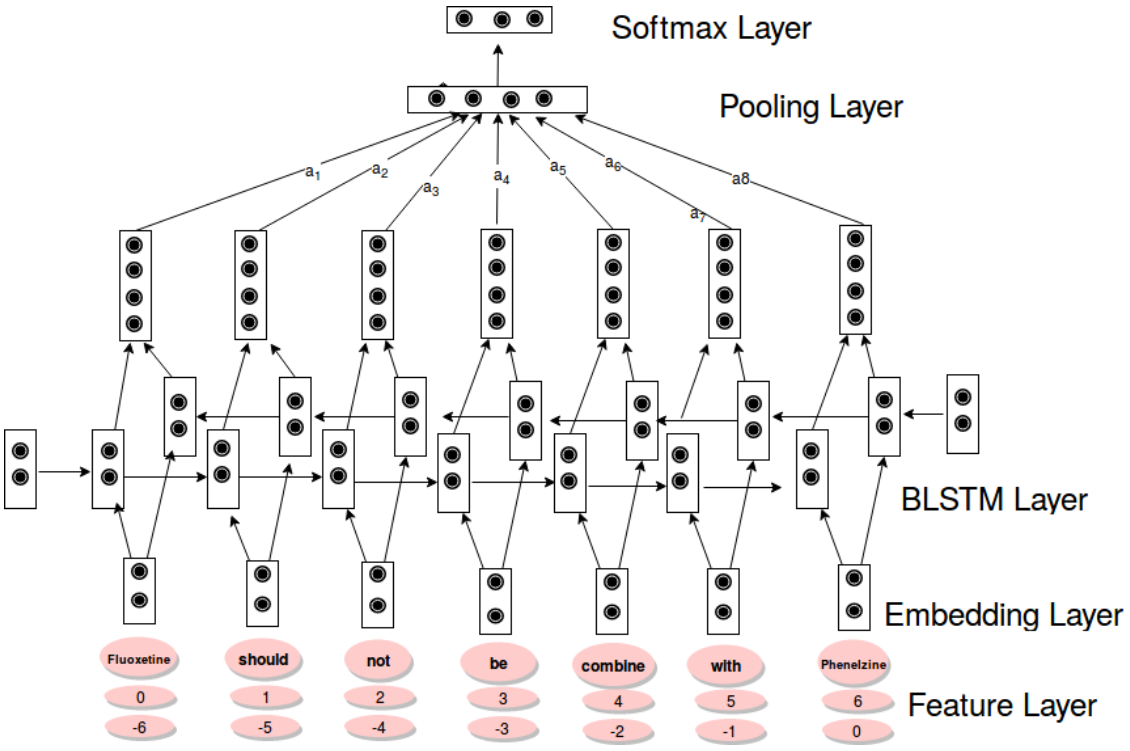


Figure 5.1: Generic Architecture of Proposed Model for Biomedical Relation Classification

Embedding Layer

In embedding layer, each discrete feature is mapped to a real-valued vector representation using a lookup or embedding matrix. Let us say M^i be the embedding matrix for i^{th} feature. Here each column of M^i is a vector for the value in i^{th} feature. Mapping can be carried out by taking the product of one hot vector of feature value and its embedding matrix. Suppose that $a_j^{(i)}$ be the one hot vector for the j^{th} feature value of the i^{th} feature then embedding layer can be obtained as follows:

$$f_j^{(i)} = M^i.a_j^{(i)} \tag{5.1}$$

$$x^i = f_1^{(i)} \oplus f_2^{(i)} \oplus f_3^{(i)} \tag{5.2}$$

Here, $\oplus$ is the concatenation operation:- thus, $x^i \in \mathbb{R}^{(n_1+....n_3)}$ is a feature vector for the i^{th} word in a sentence and n_k is the dimension of k^{th} feature. Pre-trained word vectors are used for the word embedding matrix and other feature matrices are initialized with random values.

BLSTM Layer

The output of embedding layer is a sequence of vectors for each word of the sentence. These vectors have semantic information about the individual words. In the relation classification task, important clues can be a combination of multiple words or phrases. Earlier works [Collobert et al., 2011, Sahu et al., 2016] used convolution neural network for learning features from continuous n-grams of the sentence, where n is the length of the filter. However, important clues can lie anywhere in the whole sentence and can be discontinuous. This limits the learned vectors to preserve the knowledge of discontinuous clues. To overcome this issue, we use the BLSTM network on the output of embedding layer. As we discussed, theoretically an LSTM can preserve the information of any length while computing the output. Therefore, by applying the BLSTM network on the output of embedding layer, we get a sequence of vectors each with complete knowledge of the sentence.

Pooling Layer

The idea of the pooling layer is to derive a fixed length optimal feature from variable number of word features. We experiment with two different pooling scheme types, as follows;-

(A) Max Pooling

The intuition behind using *Max* pooling is, taking one optimal over the entire sequence. BLSTM accumulates information in both forward and backward direction,

and hence each node is assumed to have complete information of a sentence. *Max* pooling takes the maximum over the entire sentence, considering all important and relevant information is accumulated in that position. Let $z_1 z_2 ... z_m$ ($z_i \in \mathbb{R}^N$) be the sequence of vectors obtained by concatenating forward, backward LSTM output of each word then:

$$z = \max_{1 \leq i \leq (m)} [z_i] \tag{5.3}$$

Where $z \in \mathbb{R}^N$ is the dimension wise max of the entire z_i's.

(B) Attentive Pooling

Taking one *max* over the entire sequence may fail to perform well when important clues for the relation are present in different clauses or lie far away in the sentence. Consider an example belonging to the *mechanism* class of interaction: *Preliminary evidence suggests that cimetidine$_{Drug}$ inhibits mebendazole$_{Drug}$ metabolism and may result in an increase in plasma concentrations of mebendazole$_{Drug}$.* The first clue *inhibits mebendazole metabolism* indicates interaction class likely to be *effect*, but the second clue *increase in plasma concentration* makes the interaction belonging to the *mechanism* class. Taking one optimal over the complete sentence may incorrectly classify this instance. To overcome this issue we use *attentive* pooling which takes optimal based on a weighted linear combination of feature vectors. Weights of the feature vectors are computed using attention mechanism which assigns weights based on the importance of that features [Bahdanau et al., 2014, Zhou et al., 2016]. The attention mechanism produces a vector α of size equal to the length of sentence. The values in this vector are the weights we would assign to each word feature vectors. Weighted linear combination of BLSTM outputs and attention weights are the output of attentive pooling layer. Let $Z \in \mathbb{R}^{N \times m}$ be the matrix of outputs obtained by BLSTM then, the output of attentive pooling would be:

$$H = tanh(Z)$$
$$\alpha = Softmax(w^{aT}H)$$
$$z = \alpha Z^T$$

where $w^a \in \mathbb{R}^N$ are learning parameters, $\alpha \in \mathbb{R}^m$ is attention weights and $z \in \mathbb{R}^N$ would be the attentive pooling layer output. The important thing to notice here is that α would be different for every sentence, *i.e.*, indicating that relevant context words may appear in different positions in different sentences.

Fully Connected and Softmax

The pooling layer output is a fixed length vector, which can be non-linearize by using *tanh* activation and then feed it to a fully connected neural network layer. In fully connected layer, we maintain a number of nodes equals to the number of classes.

$$h^3 = tanh(h^2)$$
$$p(y|x) = Softmax(W^o h^3 + b^o)$$

Here h^2 would be the pooling layer output, $W^o \in \mathbb{R}^{N \times C}$, $b^o \in \mathbb{R}^C$ are the fully connected neural network parameters, and C is the number of classes present in the task. We use the *softmax* function in the fully connected layer output to obtain a normalized probability score for each class.

5.3.1 BLSTM-RE Model

BLSTM-RE is similar to the *CNN-RE* model. Here, we use BLSTM in place of convolution neural network used in *CNN-RE* model. BLSTM-RE applies *Max*

pooling in the BLSTM output in order to derive a optimal fixed length features. *Max* pooling is obtained through Equation 5.3 for every instance. These features are then fed to fully connected neural network followed by *softmax* layer to produce final classification.

5.3.2 ABLSTM-RE Model

In the case of *ABLSTM-RE* model, we apply *attentive* pooling in the BLSTM layer output. The *attentive* pooling layer output is used as features to make a classifier by feeding this to fully connected and *softmax* layers.

5.3.3 Joint ABLSTM-RE Model

The idea of using *Joint ABLSTM-RE* is to take the advantages of both *max* and *attentive* pooling techniques. *Joint ABLSTM-RE* model uses two separate modules each with a BLSTM network. Both BLSTMs take same feature vectors as input and produce output for every word in the sentence. We apply *Max* pooling to the first and *attentive* pooling to the second BLSTM layer to get optimal features from both the modules. Concatenation of both optimal features is used for classification through fully connected and *softmax* layers.

5.4 Training and Implementation

All three models use cross entropy loss function for training the entire network. Adam optimization technique [Kingma and Ba, 2014] is used to update the parameters. Batch size of 200 is used during training of each model. Hidden layer size in *BLSTM-RE* and *ABLSTM-RE* are kept as 200, and 150 in each module of *Joint ABLSTM-RE*. For regularization, we use *dropout* [Srivastava et al., 2014] in output of max pooling layer and l_2 regularization technique. The optimal values for

Model	*Tasks*	*Dropout*	l_2 *regu.*
BLSTM-RE	DDI	0.7	0.001
	DDIC	0.7	0.001
	CRC	0.7	0.01
ABLSTM-RE	DDI	0.8	0.01
	DDIC	0.7	0.0001
	CRC	0.7	0.01
Joint ABLSTM-RE	DDI	0.9	0.01
	DDIC	1.0	0.0001
	CRC	0.9	0.001

Table 5.1: Values of different regularization parameters used in three models.

both the parameters are obtained through validation set (20% of the each training set considered as validation set) and mentioned in Table 5.1. Entire neural network parameters and feature vectors are updated while training. We implemented the proposed model in Python language using *tensorflow*[1] package. We use pre-trained 100 dimension word vectors and randomly initialized 10 dimension distance vectors for initializing embedding layer parameters.

5.5 Results and Discussion

Apart from baseline methods considered in the last chapter, here we included three more newly proposed model for comparison. All three MV-RNN[1] [Suárez-Paniagua and Segura-Bedmar, 2016], Tree-LSTM[2] [Lim et al., 2018] and Dep-LSTM[1] [Wang et al., 2017] are applied on DDIC task and use features from other NLP tools. MV-RNN[1] is a neural network based model. In particular, MV-RNN[1] uses a recursive neural network [Socher et al., 2011] for learning embedding of a sentence or a part of a sentence recursively, thereby obtaining a final vector used for classification. Dep-LSTM[1] uses the sentence dependency tree in three distinct channels to learn representation. The first channel applies BLSTM in the breadth-first search path

[1] `https://www.tensorflow.org`

of the tree, the second channel applies it in the depth-first search path, and the last channel uses it in the complete sentence. The concatenation of all the learned feature vectors is used for classification in the softmax layer. On the other hand, Tree-LSTM[2] model uses LSTM network in the parse tree of the sentence and learn representations for sentence.

5.5.1 Comparison with Baseline Methods

Table 5.2 provides a detailed comparison of our models' performance with previous approaches in all three DDI, DDIC and CRC tasks. We observe that, in all the three tasks, the *Joint ABLSTM-RE* model performance are best among all the existing feature based methods as well proposed deep learning based methods. In particular, *Joint ABLSTM-RE* model outperforms previous best feature based models by 2.08%, 3.56% and 3.66% on DDI, DDIC and CRC tasks respectively. Among the deep learning based methods, all the LSTM based models outperform the *CNN-RE* model in all the three tasks. However, in CRC task, this difference is not significant. We performed *McNemar* test to verify this.

Based on the *McNemar* test *BLSTM-RE*, *ABLSTM-RE*, and *Joint ABLSTM-RE* models outperformed *CNN-RE* model with p-values of 0.0005, 0.001 and 8.4×10^{-9} respectively in the DDIC task. As the results indicate, all LSTM based models outperformed *CNN-RE* model significantly in DDI extraction task. Among the LSTM models, *Joint ABLSTM-RE* and *BLSTM-RE* models outperformed *ABLSTM-RE* with p-values of 0.005 and 0.03 respectively on the DDIC task. Further, the *McNemar* test suggests that there is no significant difference in performance of the *Joint ABLSTM-RE* and *BLSTM-RE* models on the DDIC task. Moreover, the *McNemar* test also suggests that there is no significant difference in performance of the *CNN-RE*, *BLSTM-RE* and *ABLSTM-RE* models on the CRC task.

Tasks	*Models*	*Precision*	*Recall*	*F1 Score*
DDI	UTurku[1] [Björne et al., 2013]	85.8	58.5	69.6
	Kim[2] [Kim et al., 2015]	-	-	77.5
	NIL UCM[1] [Bokharaeian and DIAZ, 2013]	60.8	56.9	58.8
	WBI-DDI[2] [Thomas et al., 2013]	80.1	72.2	75.9
	FBK irst[2] [Chowdhury and Lavelli, 2013b]	79.4	80.6	80.0
	CNN-RE[1] [Sahu et al., 2016]	80.06	70.99	75.25
	BLSTM-RE[1]	82.23	76.60	79.32
	ABLSTM-RE[1]	80.63	75.28	77.85
	Joint ABLSTM-RE[1]	**81.23**	**81.71**	**81.67**
DDIC	UTurku[1] [Björne et al., 2013]	73.2	49.9	59.4
	UWM-TRIADS[2] [Rastegar-Mojarad et al., 2013]	43.9	50.5	47.0
	Kim[2] [Kim et al., 2015]	-	-	67.0
	NIL UCM[1] [Bokharaeian and DIAZ, 2013]	53.5	50.1	51.7
	WBI-DDI[2] [Thomas et al., 2013]	64.2	57.9	60.9
	FBK irst[2] [Chowdhury and Lavelli, 2013b]	64.6	65.6	65.1
	MV-RNN[1s] [Suárez-Paniagua and Segura-Bedmar, 2016]	52.00	48.0	50.00
	Tree-LSTM[2s] [Lim et al., 2018]	**79.30**	67.2	**72.70**
	Dep-LSTM[1s] [Wang et al., 2017]	72.53	**71.49**	72.0
	CNN-RE[1] [Sahu et al., 2016]	68.70	63.02	65.74
	BLSTM-RE[1]	70.62	66.80	68.66
	ABLSTM-RE[1]	73.34	62.41	67.43
	Joint ABLSTM-RE[1]	74.47	64.96	69.39
CRC	SVM-RE [Rink et al., 2011]	76.64	72.55	74.54
	CNN-RE [Sahu et al., 2016]	77.87	74.27	76.03
	BLSTM-RE	**79.97**	73.83	76.78
	ABLSTM-RE	78.49	74.38	76.38
	Joint ABLSTM-RE	76.45	**78.11**	**77.27**

Table 5.2: Performance comparison of our models with all baseline methods. Performance is measured based on precision, recall and f1 score. The highest scores are highlighted in bold.

Models	*Advice*	*Mechanism*	*Effect*	*Int*	*MAVG*
UTurku [Björne et al., 2013]	63.0	58.2	60.0	50.7	58.7
UWM-TRIADS [Rastegar-Mojarad et al., 2013]	53.2	44.6	44.9	42.1	47.2
Kim [Kim et al., 2015]	72.5	69.3	66.2	48.3	64.1
FBK irst [Chowdhury and Lavelli, 2013b]	69.2	67.9	62.8	**54.7**	64.8
NIL_UCM [Bokharaeian and DIAZ, 2013]	61.3	51.5	48.9	42.7	53.5
WBI-DDI [Thomas et al., 2013]	63.2	61.8	61.1	51.1	59.7
MV-RNN[1] [Suárez-Paniagua and Segura-Bedmar, 2016]	57.0	46.0	49.0	49.0	50.25
Dep-LSTM[1] [Wang et al., 2017]	**80.5**	**75.35**	**68.37**	49.0	**68.39**
CNN-RE	71.61	66.07	65.19	46.96	62.24
BLSTM-RE	75.92	72.66	65.15	47.40	65.28
ABLSTM-RE	69.68	68.06	68.28	54.16	65.04
Joint ABLSTM-RE	80.26	72.26	65.46	44.11	65.52

Table 5.3: Performance comparison between the proposed methods and top-ranking approaches on the DDIExtraction 2013 test data for DDI classification. Performance is measured through F1-Score for each class and Macro Average (MAVG). The highest scores are highlighted in bold.

Among the newly proposed deep learning based methods, Tree-LSTM model performance is 3.39 units higher than the *Joint-ABLSTM-RE* model performance in terms of F score. Similarly, Dep-LSTM model performance is 4.68 units higher than the *BLSTM-RE* model performance in terms of recall score. As discussed, Tree-LSTM model uses syntactic tree and Dep-LSTM model uses dependency tree for learning features. However, the proposed models does not use any features apart from pre-trained word vectors.

5.5.2 Class Wise Performance Analysis

We compare class wise performance of the proposed models with existing models in DDIC task (Table 5.3). We observe that *FBK irst* achieved the best performance in *Int* class. However, Dep-LSTM achieved best performance on other three

classes. *BLSTM-RE*, *ABLSTM-RE* and *Joint ABLSTM-RE* achieved the second best performance for *Mechanism*, *Effect* and *Advice* classes respectively. However, *Joint ABLSTM-RE* model performance is 2.8 units less than Dep-LSTM in terms of aggregate performance measure(macro-average F1 score). All models find it easier to detect the *Advice* interaction type compared to the instances of the other three interaction types. Similarly, all models, it most difficult to detect *Int* interaction types. The lower performance of the *Int* class can be attributed to insufficient training data. *Effect* interaction class is found to be the second most difficult class to detect by most models compared in this analysis.

5.5.3 Feature Analysis

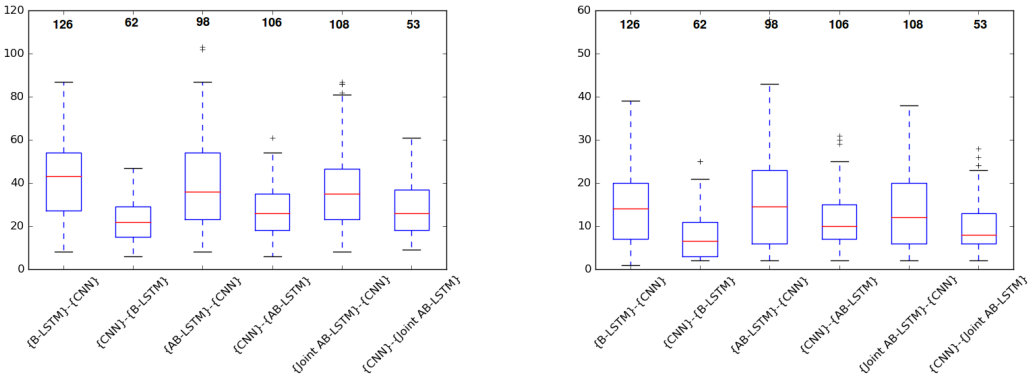
To validate the importance of each feature, we further analyzed the *Joint ABLSTM-RE* model performance by removing feature types one by one in DDIC task. It can be observed from Table 5.4 that pre-trained word embedding and position embedding are important features. About 4.6% of relative decrement is observed if the model does not use position embedding and uses random vectors for words instead of pre-trained word embedding. This indicates the importance of word embedding features. On the other hand, removal of position features reduces the performance of the model by around 1.1%.

Models	***Precision***	***Recall***	***F Score***
Joint ABLSTM-RE	74.47	64.96	69.39
Joint ABLSTM-RE - {P}	70.62	66.80	68.66
Joint ABLSTM-RE - {P+X}	71.21	61.89	66.22

Table 5.4: Contribution of each feature in **Joint ABLSTM-RE** model in DDIC task. Here P: Random Position Embedding for both P_1 and P_2, and X: Pre-trained word vector embedding.

5.5.4 LSTM vs CNN models

Earlier we discussed that intuitively it appears that LSTM models are likely to outperform CNN model on longer sentences. We perform an analysis of our results in order to determine whether or not that is the case. We investigated the length as well as entity separation between two targeted drugs of all those sentences predicted correctly by one model but incorrectly by the other in DDIC task. Figures 5.2a and 5.2b display the box plots for the sentence length and separation length between targeted drugs. Here, {X}-{Y} represent instances correctly predicted by X but not by Y. In all cases, the length represents the number of words, and numbers at the top of the boxes indicate the number of instances in that category. From the figures, we can observe that the proposed LSTM models outperformed CNN for both longer sentences and cases with greater entity separation lengths.



(a) Box plot for sentence length

(b) Box plot for entity separation

Figure 5.2: Boxplot for sentence length and entity separation length in instances. Here {X}-{Y} represent instances correctly predicted by X but not by Y. In all of these cases, length represents the number of words and numbers present at the top of the boxes represent the number of instances present in that category.

Model	Sentence Length		Entity Separation	
	True	False	True	False
CNN-RE	$26.19_{(13.38)}$	$42.51_{(21.00)}$	$11.24_{(9.39)}$	$15.17_{(12.21)}$
BLSTM-RE	$29.12_{(16.19)}$	$37.34_{(22.49)}$	$12.15_{(9.55)}$	$13.92_{(13.03)}$
ABLSTM-RE	$28.52_{(17.57)}$	$37.54_{(20.19)}$	$11.12_{(10.11)}$	$14.26_{(12.65)}$
Joint ABLSTM-RE	$28.52_{(14.50)}$	$39.97_{(21.70)}$	$12.81_{(9.49)}$	$14.19_{(11.86)}$

Table 5.5: Mean and standard deviations (in subscript) of length of sentence for True Positive and False Negative instance in DDIC task.

5.5.5 Error Analysis

In addition to the imbalance issue, we attempted to determine whether any other factors adversely affect the performance of the models. We investigated sentence length of instances that were correctly and incorrectly classified by each of the four models for the DDI classification task (Table 5.5). It can be observed that the average sentence length and entity separation length for incorrectly classified instances are always higher than in correctly classified sentences. However, LSTM based models are better than *CNN-RE* model in these cases. A further aspect of incorrectly predicted instances was the presence of multiple drug entities. A further aspect of incorrectly predicted instances is the presence of multiple drug entities in many such instances. Repetitive drug entities are more likely to behave like noise, hence, may cause neural models to lose relevant information from other words likely to be contextually important. Hence, a improved strategy is required to deal with such cases.

5.5.6 Visual Analysis

In order to confirm the ability of the model to learn attention weights based on the importance of words, we visualize the attention weights of certain of the sentences

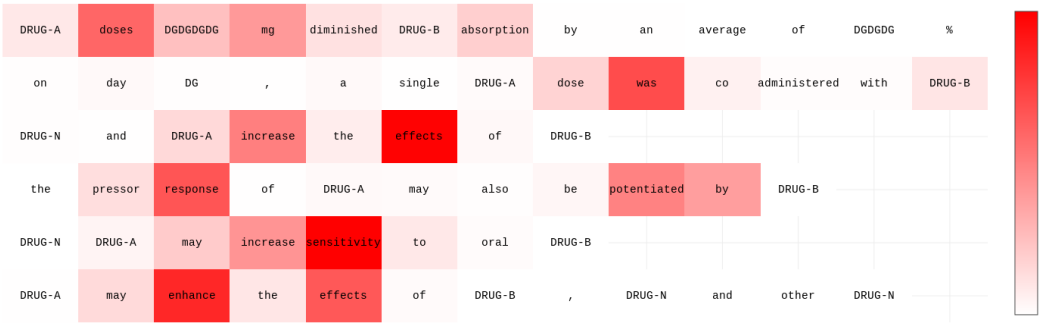


Figure 5.3: Heat map of attention weights, indicating importance of relevant words. Darkness in red color indicates relevance.

after training *Joint ABLSTM-RE*. Figure 5.3 provides the heat map of the attention weights for six instances of the test set. Here every line is a sentence with two targeted drug names replaced with the special tokens *DRUG-A* and *DRUG-B* and darkness in the red color indicate heedfulness. The figure demonstrate that our model can select important words based on the task. For example, in the sentence *DRUG-A may enhance the effects of DRUG-B , DRUG-N and other DRUG-N*, the model could effectively assign high weights to *may enhance the effects*. Similarly, in the sentence DRUG-N and DRUG-A increase the effects of DRUG-B, the model assigns high weights to the words *increase* and *effect*.

5.6 Conclusion

In this work, we have propose three LSTM based models: *BLSTM-RE*, *ABLSTM-RE* and *Joint ABLSTM-RE* for the DDI, DDIC and CRC tasks. All three models use the simple word and distance embedding as features, and learn higher level feature representation using BLSTM network. Furthermore, two of the proposed models

utilize neural attention mechanism to achieve higher level feature representation. The performances of all three models were compared to those of existing methods on SemEval-2013 DDI extraction and I2B2/VA-2010 CRC dataset. Among the three proposed models, *Joint ABLSTM-RE* model outperforms others for all the three tasks. Analysis of our results indicates that all models find it difficult to make correct predictions for drug pairs present in a long sentence and having too many other drug entries. However, if we compare between *CNN-RE* and LSTM based models, then LSTM models are generally found to have a better prediction for longer sentences than the *CNN-RE* model. We believe that new models should work in the direction of mitigating above issues to bring significant improvement.

Chapter 6

Transfer Learning for Relation Classification

6.1 Overview

In the biomedical and clinical domain, it is quite common that lack of sufficient training data does not allow to fully exploit machine learning models. However, the efficient use of transfer learning (TL) can help in utilizing knowledge learned in one task (source task) to the resource scarce task of interest (target task). In this chapter, we present three TL frameworks for relation classification tasks. We systematically investigate the effectiveness of the proposed frameworks in transferring the knowledge under multiple aspects related to source and target tasks, such as similarity or relatedness between the source and target tasks, and size of training data for source task. Our empirical results show that the proposed frameworks, in general, improve the model performance. However, these improvements do depend on aspects related to source and target tasks. This dependence then finally determine the choice of a particular TL framework.

6.2 Introduction

Recurrent neural networks and their variants, such as LSTM network, have shown to be effective models for many natural language processing tasks [Mikolov et al., 2010, Graves, 2013, Karpathy and Fei-Fei, 2014, Zhang and Wang, 2015, Chiu and Nichols, 2015, Zhou et al., 2016] including the tasks presented in earlier chapters. However, the requirement of huge gold standard labeled datasets for training makes it difficult to apply them to task for which few such resources exist, which is often the case in the biomedical domain. In the biomedical field, obtaining labeled data is not only time consuming and costly but also requires domain knowledge. TL has been used successfully in such cases across multiple domains. Transfer learning aims to apply the knowledge gained while training a model for a Task-A (*Source Task*), where we have sufficient gold standard labeled data to a different Task-B (*Target Task*) where we do not have enough training data [Pan and Yang, 2010]. In literature, various TL frameworks have been proposed [Pan and Yang, 2010, Mou et al., 2016, Yosinski et al., 2014]. With the recent surge in applications of TL using neural network based models in computer vision and image processing [Yosinski et al., 2014, Azizpour et al., 2015] as well as in NLP [Mou et al., 2016, Zoph et al., 2016, Yang et al., 2017], this work explores TL frameworks using a neural models for relation classification in the biomedical domain.

A very common approach to applying TL is to train learning models on source and target tasks in sequence. We refer to this approach as *sequential TL*. Furthermore, if there exists a bijection mapping between the label sets of source and target tasks, then the entire model trained on the source task can be transferred to the target task. Otherwise, only a partial model can be transfered. In this work, existence of a bijection mapping between two label sets is also called as *same label set* and nonexistence of bijection mapping as *disparate label set*. In NLP, transfer of feature representations is the most common form of partial model transfer. Instead

of performing the training in a sequential manner, an alternative method is to train the model on both source and target data simultaneously [Yang et al., 2017]. This is very similar to *multi-task learning* MTL [Collobert and Weston, 2008]. This way of simultaneous training can be carried out in multiple ways. These options make it possible to design several variants of the TL framework.

In addition the options of using training data in different ways, using partial or complete model transfer, and presence or absence of bijection mapping between two label sets, other aspects such as *selection of the source task*, *its size* and *relatedness or similarity to the target task* determine the selection of the most relevant TL model. Intuitively, it is preferable for the source task to be as similar as possible to the target task. For example, if the target task concerns the binary classification of drug-drug interactions (DDIs) mentioned in social media text or in doctors' notes, then a potentially suitable source task would be the binary classification of DDI mentioned in research articles. Here, the difference lies in the nature of texts appearing in the two corpora. In the case of doctors' notes, the text is likely to be short and precise compared to the research articles. In other words, the feature spaces representing data for the source and target tasks differ from each other, although the two label sets are same. On the other hand, it is also possible that there does not exist any bijection between the labels of the source and target tasks. An example of such a scenario would be if the target task required multi-class classification of DDIs, while the source task only concerned binary classification.

According to the various possible scenarios introduced in the above discussion, we present three different TL frameworks in this study. Our motivation is to systematically explore various TL frameworks for the task of relation classification in the biomedical domain and try to empirically analyze the results we obtain. Our contribution in this chapter can be summarized as follows:

- We present and evaluate three TL framework variants based on LSTM models

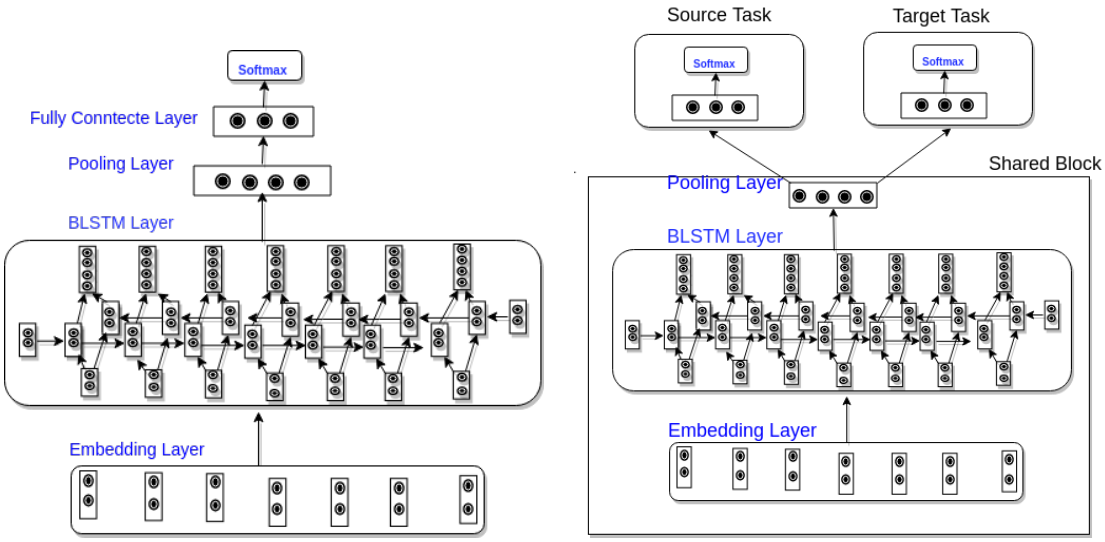
for different relation classification tasks in biomedical and clinical text.

- We analyze the impact of relatedness (implicit or explicit) between the source and target tasks on the effectiveness of TL framework.
- We explore how the size of the training data corresponding to source task impacts on the effectiveness of TL frameworks.

6.3 Model Architectures

In this section, we explain the three TL frameworks using the BLSTM-RE architecture. BLSTM-RE is the same model as discussed in the previous chapter. BLSTM-RE model is briefly summarized for the sake of completeness. We assume that positions of the two entities of interest, referred to as target entities, within the sentence are known.

The generic neural network architecture for the relation classification task consist of the following layers: *word level feature layer*, *embedding layer*, *sentence level feature extraction layer*, *fully connected and softmax layers*. We define features for all words in the *word level feature layer*, which also includes some features relative to the two targeted entities. In the *embedding layer* every feature gets mapped to a vector representation through a corresponding embedding matrix. Raw features are combined with the entire sentence and a fixed length feature representation is obtained in the *sentence level feature extraction layer*. Although a CNN or other variants of the RNN can be used in this layer, we use BLSTM because of its relatively better ability to take into account discontiguous features. The *Fully connected and softmax layer* map sentence level feature vectors to a class probability. In summary, the input for these models would be a sentence with the two targeted entities and the output would be a probability distribution over each possible relation class between them.



(a) BLSTM-RE Model (b) T-BLSTM-Multi Model

Figure 6.1: Proposed Model Architecture: BLSTM is bi-directional long short term memory network

BLSTM-RE

Suppose $w_1 w_2 .....w_m$ is a sentence of length m. Two targeted entities e_1 and e_2 correspond to some words (or phrases) w_i and w_j respectively. In this work we use word and its position from both targeted entities as features in the word level feature layer. Positional features are important for relation classification task because they inform the model to know the targeted entities [Collobert et al., 2011]. The output of *embedding layer* would be a sequence of vectors $x_1 x_2 .....x_m$ where $x_i \in \mathbb{R}^{(d_1+d_2+d_3)}$ is the concatenation of word and position vectors. d_1,d_2 and d_3 are the embedding dimensions corresponding to the word, the position from first entity and the position from second entity features respectively. We use a BLSTM with *max* pooling in the *sentence level feature extraction layer*. This layer is responsible for obtaining an optimal fixed length feature vector from entire sentence. The basic architecture is shown in the figure 6.1a.

T-BLSTM-Mixed

T-BLSTM-Mixed is a specific way to use the *BLSTM-RE* model in a transfer learning framework. In this case, instances from both source and target tasks are fed into the same *BLSTM-RE* model. While training, we pick one batch of data from the source or target in a random order with equal probability. Since training happens simultaneously for both the source and target dataset, we can say that model will learn features which are applicable to both dataset. It is quite obvious that this model is only applicable for those cases in which bijection mapping between labels of the source and target tasks exists.

T-BLSTM-Seq

The convergence of neural network based models depends on the initialization of model parameters. Several studies [Hinton et al., 2006, Bengio et al., 2007, Collobert et al., 2011] have shown that initializing parameters with values from other supervised or unsupervised pre-trained models often improves the model convergence. In this framework, we firstly train our model with the source task dataset and use the learned parameters to initialize the model parameters for training a separate model to carry out the target task. We call this framework *T-BLSTM-Seq*. *T-BLSTM-Seq* can be applicable for the transfer of both the *same label set* and *disparate label* sets. We transfer the entire set of network parameters if there exists a bijection mapping between the source and target label sets. Otherwise, we only share model parameters up to the second last layer of the network. The left out last layer is randomly initialized.

T-BLSTM-Multi

We propose another transfer learning framework, called *T-BLSTM-Multi*, using the same backbone of *BLSTM-RE* model. As shown in figure 6.1b, this model has two *fully connected* and *softmax* layers: one for the source task and other is for the target task. The other layers of the models are shared for the two tasks. While training, the parameters of the shared block are updated with training instances from both source and target data and the *fully connected* layer is updated only with its corresponding task data. A batch of instances is picked in a similar manner to *T-BLSTM-Mixed*. This method of training is also called *multi-task learning*, but in that case, the focus is on the performance of both the source and target tasks. The *T-BLSTM-Multi* model is also applicable for both *disparate label set* as well as *same label set* transfer.

Training and Implementation

Pre-trained word vectors are used to initialize the word embeddings, and random vectors are used for other feature embeddings. We use *GloVe* [Pennington et al., 2014] on PubMed corpus to obtain word vectors. The dimensions of word and position embeddings are set to 100 and 10, respectively. Adam optimization [Kingma and Ba, 2014] is used for training all models. All parameters, *i.e.*, word embeddings, position embeddings and network parameters are updated during training. We fixed batch size to 100 for all the experiments. In the case of *T-BLSTM-Mixed* and *T-BLSTM-Multi* one of the source and target task is chosen with equal probability and one batch of instances from the corresponding training set are then picked. All the remaining hyperparameters are set according to [Sahu and Anand, 2017a]. The entire implementation is carried out in the Python language using the *tensorflow*[1] library.

[1] `https://www.tensorflow.org/`

6.4 Task Definitions and Used Datasets

In this section, we briefly describe the tasks and corresponding datasets used in this study. Apart from the three tasks (DDI, DDIC and CRC), described in the previous chapter, two more tasks are considered in this study. Adverse Drug Event (ADE) extraction and Event Argument Extraction (EAE) are the two new tasks. We present description of all tasks for the sake of completeness.

Drug Drug Interaction Extraction (DDI): Identifying DDIs present in the text is a kind of relation classification task - given two drugs or pharmacological substances that are mentioned in a sentence, it is necessary to determine whether or not there is an interaction between them. .

Drug Drug Interaction Class Extraction (DDIC): DDI can appear in the text with different semantic senses, which we call as DDI class. In case of DDIC, we need to identity exact class of interactions among drugs in a sentence. The SemEval 2013[2] DDI Extraction task had four kinds of interaction, i.e., *Advice, Effect, Mechanism* and *Int.*

Clinical Relation Classification (CRC): Clinical relation classification is the task of identifying relation among clinical entities such as *Problem, Treatment and Test* in clinical notes or discharge summaries. In *Ex.1*, *allergic* and *rash* have *medical problem indicate medical problem* relation.

Ex.1: *She is allergic*$_{Problem}$ *to augmentin which gives her a rash*$_{Problem}$.

Adverse Drug Event Extraction (ADE): Adverse drug events occur when an adverse effect happens due to consumption of a drug. In NLP, ADE extraction is the process of extracting adverse relations between a drug and a condition or disease in text. For instance in *Ex.2*, the treating of the patient suffering from *thyrotoxicosis* disease with *methimazole* led to an adverse effect.

[2] `https://www.cs.york.ac.uk/semeval-2013/task9/`

Ex.2: *A 43 year old woman who was treated for thyrotoxicosis*$_{Disease}$ *with methimazole*$_{Drug}$ *developed agranulocytosis*

Event Argument Extraction (EAE): In the biomedical domain, events are broadly described as a change in the state of a bio-molecule or bio-molecules [Pyysalo et al., 2012]. Every events has its own set of arguments and EAE is the task of identifying all arguments of an event and their roles. In this task, entities and triggers (representing event occurrences) are provided in each sentence and the task is to find the role (relation) between all pairs of triggers and entities. For this work, we do not differentiate between different types of role. This implies that if an entity is an argument of a trigger, then there is a positive relation between them, otherwise there is no relation. For instance, in *Ex.3 (reptin, regulates)*, *(regulates, growth)* and *(growth, heart)* represent positive relation.

Ex.3: *Reptin*$_{Protein}$ *regulates*$_{Regulation}$ *the growth*$_{Growth}$ *of the heart* $_{Organ}$.

Source Data

Below we summarize datasets for source tasks. Statistics of all source datasets are shown in Table 6.1.

BankDDI: This is a set of documents from DrugBank[3] that has been manually annotated with DDI relations. DrugBank contains drug information in the form of documents which have been curated by accredited experts. We separated this from the complete dataset of SemEval-2013 DDI extraction challenge dataset for our experiments.

BankDDIC: This dataset is same as BankDDI, with the additional of class labels for the DDI relations [Segura-Bedmar et al., 2013].

ADE: For ADE extraction, we used the dataset described in [Gurulingappa et al., 2012b, Gurulingappa et al., 2012a]. The shared dataset contains manually

[3] `https://www.drugbank.ca/`

annotated adverse drug events mentioned in a corpus of Medline abstracts.

Task	Corpus	Training Set	Test Set
BankDDI	Pairs	14176	3694
	Positive DDIs	3617	884
	Negative DDIs	11559	2810
BankDDIC	Pairs	14176	3694
	Negative DDIs	11559	2810
	Effect	1471	298
	Mechanism	1203	278
	Advice	813	214
	Int	130	94
ADE	Pairs	8867	3802
	Positive ADEs	4177	1791
	Negative ADEs	4690	2011
EAE	Pairs	21594	11443
	Positive EAEs	4492	2202
	Negative EAEs	17102	9241
CRC	Pairs	43602	18690
	Negative CRCs	36324	15995
	TeRP	2136	915
	TrAP	1832	784
	PIP	1541	660
	TrCP	368	157
	TeCP	353	150

Table 6.1: Statistics of **source task** datasets

EAE: We used the MLEE[4] corpus for event argument identification [Pyysalo et al., 2012]. the MLEE dataset has 20 types of events trigger and 11 entity types for relation classification.

CRC: For clinical relation classification, we used the dataset from I2B2/VA 2010[5] clinical information extraction challenge [Uzuner et al., 2011]. Similarly to earlier chapters, we consider $TrCP$, $TrAP$, PIP, $TeRP$ and $TeCP$ classes in this

[4] `http://nactem.ac.uk/MLEE/`

[5] `https://www.i2b2.org/NLP/Relations/`

case.

Target Data

Below we summarize datasets for the target tasks. Statistics of the datasets are shown in Table 6.2.

Task	Corpus	Training Set	Test Set
MedDDI	Pairs	1319	334
	Positive DDIs	227	95
	Negative DDIs	1092	239
MedDDIC	Pairs	1319	334
	Negative DDIs	1092	239
	Effect	149	62
	Mechanism	61	24
	Advice	7	7
	Int	10	2
CRC_5	Pairs	2125	18690
	Negative CRCs	1816	15885
	TeRP	106	915
	TrAP	91	784
	PIP	77	660
	TrCP	18	157
	TeCP	17	150

Table 6.2: Statistics of **target task** dataset

MedDDI: MedDDI is a set of MedLine abstracts that have been manually annotated with DDI relations. The dataset was released as part of SemEval-2013 DDI extraction challenge. MedDDI differs from BankDDI in several ways, since: MedDDI consists of MedLine abstracts whereas BankDDI consist of DrugBank documents. Since MedLine abstract form an integral part of research articles, they typically contain many technical terms and use long sentences. In contrast, DrugBank documents consist of concise sentences written by medical practitioners,

which are relatively shorter and more easily comprehensible than those found in MedLine abstracts.

MedDDIC: This is the same dataset as MedDDI, but with semantic labels that denote the exact class of the annotated DDI relations. Both MedDDI and MedDDIC datasets were used in SemEval-2013 DDI extraction challenge.

CRC$_5$: In this case we take a 5% subset of each class in the full CRC training dataset and considered that to be the training set. The test set remains the same.

Preprocessing

We use the same preprocessing strategies for all datasets. Pre-processing steps include: the conversion of all words into lower case form, the tokenization of sentence using geniatagger[6], and the replacement of digits with the *DG* symbol. Furthermore, if any sentence has more than two entities, we create a separate instance for each pair of entities. In all sentences, the two targeted entities were replaced with their types and offset position. For example, the sentence in *Ex.1* will become *She is ProblemA to augmentin which gives her a ProblemB*. We removed few negative instances in BankDDI and MedDDI datasets based on the similar set of rules as used in [Sahu and Anand, 2017a, Zhao et al., 2016].

6.5 Results and Discussion

Firstly, we discuss our experimental design to evaluate performance of the three TL frameworks using various settings. Subsequently, we analyze and discuss the results that we obtained. We treat the performance of the *BLSTM-RE* model on the target task as baseline. In the baseline experiments, training was carried out on the training set of each of the three target datasets and performance on the

[6] `http://www.nactem.ac.uk/GENIA/tagger/`

respective test sets are reported in Table 6.3.

Task	Precision	Recall	F Score
MedDDI	$0.561_{(0.03)}$	$0.431_{(0.03)}$	$0.488_{(0.02)}$
MedDDIC	$0.684_{(0.08)}$	$0.273_{(0.01)}$	$0.390_{(0.03)}$
CRC_5	$0.529_{(0.04)}$	$0.492_{(0.01)}$	$0.510_{(0.009)}$

Table 6.3: Baseline Performance: Results of **BLSTM-RE** model applied on three different target tasks. Figures in the Precision, Recall and F Score column indicate result corresponding to best F1 Score and subscripts shows the standard deviation of five runs of model.

The baseline model use pre-trained word embeddings, a form of unsupervised transfer learning framework. As we have shown the superior performance of such models using pre-trained vectors than using random vectors, we do not perform any experiment related to that. Five runs with different random initialization were taken for each model, and the best results in terms of F1-score along with corresponding precision and recall are shown in Tables. We experiment with different combinations of source and target tasks to analyze the effects of similarity between feature spaces corresponding to the source and target tasks as well as their label sets on the choice of TL frameworks.

6.5.1 Performance on Same Label Set Transfer

We firstly examine the relative improvement of various TL models over the baseline results on the DDI and DDIC tasks. Table 6.4 shows the performance of all TL models on these two tasks under various settings. *Type* in Table 6.4 indicates the degree of semantic relatedness between the source and target tasks. For example, annotations in both the *BankDDI* and *MedDDI* dataset denote drug-drug interactions, and hence are of the similar semantic type. However, the *EAE* dataset

Type	Model	Precision	Recall	F Score	Δ
Similar	*T-BLSTM-Mixed*$_{(BankDDI \Rightarrow MedDDI)}$	$0.656_{(0.02)}$	$0.705_{(0.03)}$	$0.680_{(0.03)}$	39.34%
	T-BLSTM-Seq$_{(BankDDI \Rightarrow MedDDI)}$	$0.678_{(0.02)}$	$0.621_{(0.03)}$	$0.648_{(0.03)}$	32.78%
	T-BLSTM-Multi$_{(BankDDI \Rightarrow MedDDI)}$	$0.701_{(0.05)}$	$0.568_{(0.05)}$	$0.627_{(0.02)}$	28.48%
	T-BLSTM-Mixed$_{(BankDDIC \Rightarrow MedDDIC)}$	$0.631_{(0.04)}$	$0.505_{(0.02)}$	$0.561_{(0.01)}$	43.84%
	T-BLSTM-Seq$_{(BankDDIC \Rightarrow MedDDIC)}$	$0.600_{(0.03)}$	$0.463_{(0.02)}$	$0.550_{(0.01)}$	41.02%
	T-BLSTM-Multi$_{(BankDDIC \Rightarrow MedDDIC)}$	$0.579_{(0.01)}$	$0.421_{(0.006)}$	$0.487_{(0.006)}$	24.87%
Dissimilar	*T-BLSTM-Mixed*$_{(ADE \Rightarrow MedDDI)}$	$0.494_{(0.02)}$	$0.515_{(0.03)}$	$0.505_{(0.02)}$	3.48%
	T-BLSTM-Seq$_{(ADE \Rightarrow MedDDI)}$	$0.595_{(0.02)}$	$0.294_{(0.03)}$	$0.394_{(0.02)}$	-19.26%
	T-BLSTM-Multi$_{(ADE \Rightarrow MedDDI)}$	$0.533_{(0.02)}$	$0.505_{(0.02)}$	$0.518_{(0.01)}$	6.14%
	T-BLSTM-Mixed$_{(EAE \Rightarrow MedDDI)}$	$0.540_{(0.03)}$	$0.557_{(0.05)}$	$0.549_{(0.02)}$	12.5%
	T-BLSTM-Seq$_{(EAE \Rightarrow MedDDI)}$	$0.544_{(0.03)}$	$0.515_{(0.04)}$	$0.529_{(0.02)}$	8.40%
	T-BLSTM-Multi$_{(EAE \Rightarrow MedDDI))}$	$0.538_{(0.02)}$	$0.589_{(0.05)}$	$0.562_{(0.02)}$	15.16%

Table 6.4: Results of TL frameworks applied to **same label set transfer**. Here $(X \Rightarrow Y)$ indicates transfer from X dataset to Y dataset and *Type* indicates similarity between the source and target dataset. Figures in the Precision, Recall and F Score columns indicate results corresponding to best F1 Score and subscripts denote standard deviation of five runs of model. Δ shows the relative percentage improvement over the baseline (without TL) method

provides trigger-arguments relations, which are not of the same semantic type as drug-drug interactions, although both tasks corresponds to binary classification task. As the results indicate, the *T-BLSTM-Mixed* model gave the best performance (in terms of F1-score) for the *similar* type tasks, whereas *T-BLSTM-Multi* produced the worst performance. However, all of the TL models produced significant improvements over the baseline results. *T-BLSTM-Mixed* obtained approximately 40% relative improvement over the baseline for the DDI task and approximately 44% for the DDIC task. On the other hand, *T-BLSTM-Multi* produced the best performance for the *dissimilar* type tasks and *T-BLSTM-Seq* produced the worst. In fact, *T-BLSTM-Seq* produced poorer performance than the baseline in one case.

Source⇒Target	*T-BLSTM-Seq*			*T-BLSTM-Multi*		
	Precision	Recall	F Score	Precision	Recall	F Score
BankDDI⇒MedDDIC	$0.50_{(0.08)}$	$0.378_{(0.03)}$	$0.431_{(0.008)}$	$0.603_{(0.05)}$	$0.368_{(0.03)}$	$0.457_{(0.03)}$
CRC⇒MedDDIC	$0.448_{(0.05)}$	$0.368_{(0.03)}$	$0.404_{(0.02)}$	$0.468_{(0.02)}$	$0.389_{(0.02)}$	$0.425_{(0.01)}$
EAE⇒MedDDIC	$0.596_{(0.04)}$	$0.326_{(0.03)}$	$0.421_{(0.02)}$	$0.488_{(0.03)}$	$0.452_{(0.06)}$	$0.469_{(0.04)}$
ADE⇒MedDDIC	$0.512_{(0.06)}$	$0.221_{(0.01)}$	$0.308_{(0.01)}$	$0.447_{(0.04)}$	$0.400_{(0.03)}$	$0.422_{(0.02)}$
BankDDI⇒CRC_5	$0.555_{(0.02)}$	$0.485_{(0.01)}$	$0.518_{(0.01)}$	$0.546_{(0.01)}$	$0.508_{(0.02)}$	$0.526_{(0.006)}$
BankDDIC⇒CRC_5	$0.564_{(0.04)}$	$0.447_{(0.02)}$	$0.498_{(0.01)}$	$0.523_{(0.01)}$	$0.557_{(0.02)}$	$0.539_{(0.007)}$
EAE⇒CRC_5	$0.543_{(0.02)}$	$0.533_{(0.02)}$	$0.538_{(0.006)}$	$0.587_{(0.01)}$	$0.548_{(0.01)}$	$0.567_{(0.01)}$
ADE⇒CRC_5	$0.516_{(0.003)}$	$0.503_{(0.01)}$	$0.509_{(0.006)}$	$0.598_{(0.03)}$	$0.483_{(0.02)}$	$0.535_{(0.007)}$
BankDDIC⇒MedDDI	$0.605_{(0.04)}$	$0.452_{(0.03)}$	$0.518_{(0.01)}$	$0.623_{(0.04)}$	$0.557_{(0.03)}$	$0.588_{(0.02)}$
CRC⇒MedDDI	$0.569_{(0.11)}$	$0.473_{(0.17)}$	$0.517_{(0.03)}$	$0.631_{(0.05)}$	$0.505_{(0.02)}$	$0.561_{(0.02)}$

Table 6.5: Results of *T-BLSTM-Seq* and *T-BLSTM-Multi* on **disparate label set transfer** task. Here $(X \Rightarrow Y)$ indicates transfer from X dataset to Y dataset. Figures in Precision, Recall and F Score columns indicate results corresponding to the best F1 Score and subscripts denote standard deviation of five runs of model.

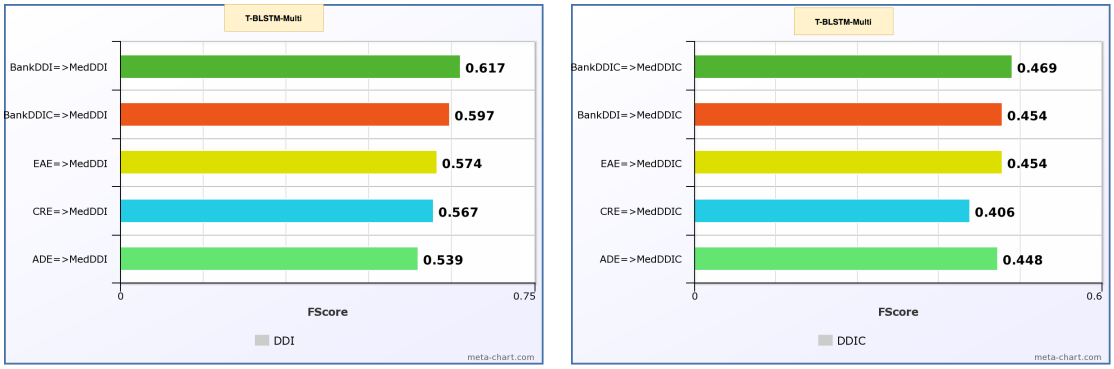
6.5.2 Performance on Disparate Label Set Transfer

Next, we examine the performance of relevant TL models when there does not exist a bijection mapping between the two label sets corresponding to the source and target tasks (Table 6.5). As the *T-BLSTM-Mixed*, by design, requires the existence of a bijection mapping between the two label sets, we exclude this model for this case. Among the remaining two models, *T-BLSTM-Multi* always led to significantly improved performance compared to the respective baseline results. On the other hand, the performance of the *T-BLSTM-Seq* model is rather inconsistent, especially when *ADE* was used as source data.

6.5.3 Analyzing Similarity between Source and Target Tasks

Earlier, we observed that the level of similarity between the source and target tasks affects the relative performance of each TL framework. Here, we try to analyze the reason for this observation. *T-BLSTM-Mixed* and *T-BLSTM-Seq* models transfer full knowledge, or in other words, in both cased, the complete model is shared between the source and target tasks. This allows the last layers to see a higher number of examples and to be adaptive to samples from both the source and target data. On the other hand, *T-BLSTM-Multi* only shares the partial model up to the second last layers. In this case, the last layers for the source and target tasks are trained separately and are not being shared between the two. Thus, the last layers are specific to the respective tasks. When there is similarity between the source and target tasks, as well as the existence of a bijection mapping, target tasks benefit from sharing the full model. In such scenarios, co-training seems better suited, as *T-BLSTM-Mixed* was found to be the best among the three frameworks. On the other hand, *T-BLSTM-Multi* fails to exploit the full knowledge present in the training data of the source task. However, this becomes advantageous for the *T-BLSTM-Multi* framework, in the case of an absence of bijection between the source and target label sets. The last layer, in *T-BLSTM-Multi*, takes the shared knowledge and tunes it to the specific target task. This observation also fits well with the observations made in [Yosinski et al., 2014] which note that the initial layers are relatively generic and become more specific as we move towards the last layer.

Variability amongst the different sizes of the source data training could have influenced the observed performance differences. Therefore, in order to remove this potential effect all source dataset were made to be of the same size (number of training instances = 8867) as the smallest of all source training datasets. During random selection, the proportion of instances from all classes were maintained to be the same as in the original set. We show only the results for *T-BLSTM-Multi*



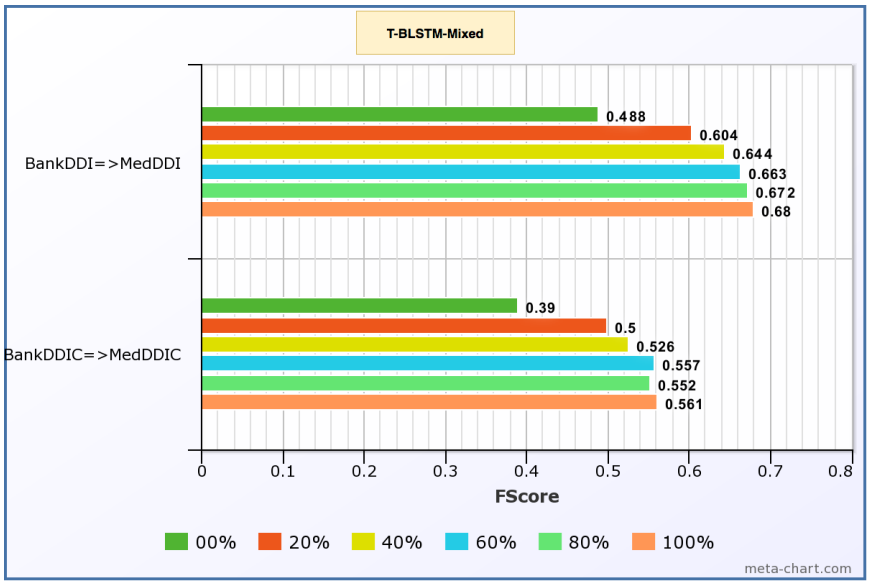
(a) T-BLSTM-Multi Model (DDI task) (b) T-BLSTM-Multi Model (DDIC task)

Figure 6.2: Performance of proposed models with different source task on same size data

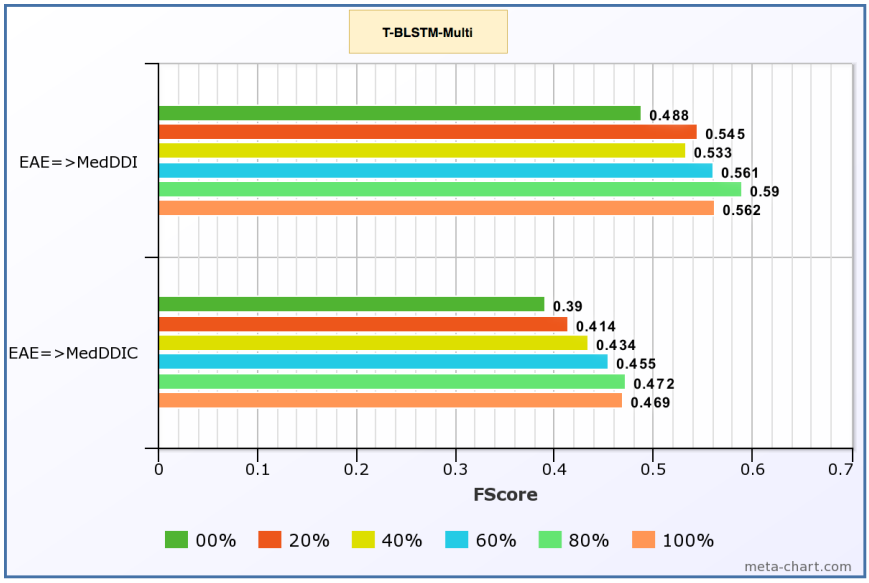
in Figure 6.2. However, similar results are also obtained for the other models. We observe that the performance obtained when using different source dataset of the same size with the performance obtained using the same set of source data, but of different sizes. This indicates that context and label mapping played the more crucial roles than the size of the selected source data.

6.5.4 Analyzing Size of Source Task Dataset

One of the strongest arguments generally given in favour of the use of transfer learning is datasets of an insufficient size hinder the performance of learning algorithms applied to the target task. Performance can be enhanced by utilizing information available in other, larger datasets. In this section, we investigate the effect of the size of source data on the performance improvement of the *T-BLSTM-Mixed* and *T-BLSTM-Multi* models. Figure 6.3 shows the results for both similar and dissimilar tasks. In both scenarios, even having 20% of the complete source data significantly improves the performance. However, for similar tasks, there is a trend towards consistent improvement as the amount of source data is



(a) T-BLSTM-Mixed: transfer on similar task



(b) T-BLSTM-Multi: transfer on dissimilar task

Figure 6.3: Performance graph of proposed model as the size if the source task dataset is varied

Models	**DDIC**	**DDI**
FBK-Irst [Chowdhury and Lavelli, 2013b]	0.398	0.530
SCAI [Bobic et al., 2013]	0.420	0.47
WBI [Thomas et al., 2013]	0.365	0.503
UTurku [Björne et al., 2013]	0.286	0.479
UMAD [Rastegar-Mojarad et al., 2013]	0.312	0.479
BLSTM-RE	0.390	0.488
T-BLSTM-Multi$_{EAE=>\odot}$	**0.469**	**0.562**
T-BLSTM-Multi$_{CRC=>\odot}$	0.425	0.561
T-BLSTM-Multi$_{ADE=>\odot}$	0.422	0.518

Table 6.6: Performance comparison of existing methods for DDI and DDIC tasks. Here values represent the **F1 Scores** for each task. $\odot$ is MedDDI or MedDDIC

increased. In contrast, for dissimilar task, there was a certain amount of fluctuation in performance as the amount of source data was increased. A possible reason for the fluctuation is the use of too much source data may confuse the model.

6.5.5 Comparison with state-of-art Results

As a final step, we compare our results with the state-of-the-art results obtained for the target tasks of *DDI* and *DDIC*. Table 6.6 shows the best results reported by teams that participated in the SemEval 2013 DDI extraction challenge [Segura-Bedmar et al., 2013] as well as the results obtained by our *BLSTM-RE* and *T-BLSTM-Mixed* models on dissimilar tasks. We can observe that *BLSTM-RE* cannot outperform the results of the best performing teams in the original challenge. However, using the TL framework, *T-BLSTM-Multi* is able to improve upon the previous state-of-the-art systems, even for dissimilar tasks.

6.6 Conclusions

In this work, we have presented various transfer learning frameworks based on BLSTM-RE model for relation classification task in the biomedical domain. We observe that, in general transfer learning does help in improving the performance. However, the level of similarity between the source target task, as well as the size of the corresponding source data affects the performance and hence plays an important role in the selection of an appropriate TL framework.

Chapter 7

Conclusion and Future Directions

In this thesis, firstly, we presented a unified model for drug, disease and clinical entity recognition tasks. The presented model, called *CWBLSTM*, uses BLSTMs in a hierarchy to learn better feature representations and uses CRF to infer the correct labels for each word in the sentence simultaneously. The proposed model outperforms task-specific as well as task-independent baselines in all the three tasks. Through various analyses, we demonstrated the importance of each component and the features used by the model. Our analyses suggest that pre-trained word embeddings and character-based word embeddings play complementary roles to learning appropriate representations. The embeddings along with the incorporation of tag dependencies, are important ingredients in improving the performance of drug, disease and clinical entity recognition tasks.

Next, we analyzed four models namely, *CNN-RE, BLSTM-RE, ABLSTM-RE* and *Joint ABLSTM-RE* for relation classification tasks in biomedical and clinical texts. All the four models use word and distance embeddings as features and learn higher-level feature representations using a CNN or BLSTM network. ABLSTM-RE and Joint ABLSTM-RE also utilize neural attention mechanisms to obtain higher level feature representations. The performance of the all four models was compared

with existing methods on SemEval-2013 DDI extraction and I2B2/VA-2010 CRE datasets. The *Joint ABLSTM-RE* model achieves significantly better performance in all the three tasks. Analysis of the results indicates the following important points: imbalance and noise adversely affect all models, the *Advice* interaction class is the easiest to predict in the DDIC task, multiple mentions of other drug names negatively effect all models, models are more likely to make incorrect classification for longer sentences and LSTM-based models are generally found to have better performance for longer sentence in comparison to the *CNN-RE* model.

Finally, we presented three TL methods, namely, *T-BLSTM-Mixed*, *T-BLSTM-Multi* and *T-BLSTM-Seq* which transfer knowledge from the dataset of one task, for which there is sufficient gold standard labeled data, into the other task, where we do not have enough training data. All the three proposed frameworks use a *BLSTM-RE* model to obtain improved performance on various biomedical and clinical relation classification tasks. We observe that, in general transfer learning does help to improve the performance. However, the degree of similarity between source task and the target task, as well as the size of corresponding source data, affects the performance and hence plays important role in the selection of an appropriate transfer learning method.

Future Work

Below, we describe a number of ways in which the present work could be extended:

Although the frameworks that we have proposed reduce the requirement for explicit feature engineering, it is at the expense of the loss of an interpretable learned feature representation. In the computer vision domain, there have been some impressive works in the line of interpreting features [Yosinski et al., 2015, Liu et al., 2016a], but in the NLP field, only a few such exploratory works have been reported [Karpathy et al., 2015, Li et al., 2016]. Hence, building a model which can also

provide an interpretable representation could be an immediate goal of subsequent work.

In the transfer learning framework, our work was limited to exploring different relation classification tasks [Sahu and Anand, 2018]. However, it would be possible to use this framework for various named entity recognition tasks and a combination of entity and relation classification together. As future work, we would like to explore this direction.

In this thesis, we studied different deep learning based models for binary relation classification tasks in biomedical and clinical texts [Sahu et al., 2016, Sahu and Anand, 2017a, Raj et al., 2017]. However, higher order relationships such as complex events are of more vital importance in the biomedical domain. In the future, therefore, it will be important to systematically study the application of deep learning methods to higher order relation classification tasks.

Glossary

CNN CNN stands for convolution neural network. It is a kind of feed-forward network.. ix, xvii, 11, 12, 18, 55, 57, 77, 115

CRF CRF stands for conditional random field. It is a kind of graphical model that can use for structure prediction.. ix, 2, 8, 10, 33, 34, 36, 40, 48, 49, 52, 115

DDI DDI stands for drug-drug interaction. It is a situation in which a drug affects the activity of another drug when both are administered together.. 1, 65, 79, 93, 97, 102

GloVe GloVe stands for global vector. It is a word embedding technique. It uses a large corpus and learn embeddings through preserving co-occurrence knowledge of words.. xi, 15, 28, 31, 41, 58, 60, 67

HMM HMM stands for hidden markov model. It is a generative model that can be use for structure prediction.. 8

LSTM LSTM stands for long short-term memory network. It is class of RNN network. LSTM uses a memory cell and three gates to maintain past information. viii, ix, xvii, 12, 21, 38, 51, 87, 93, 115

MTL Multi-task learning (MTL) is a subfield of machine learning in which multiple learning tasks are solved at the same time, while exploiting commonalities and

differences across tasks.. 97

PubMed PubMed is a free search engine accessing primarily the MEDLINE database of references and abstracts on life sciences and biomedical topics. . x, 1, 29, 41, 42, 58, 60, 67, 101

RNN RNN stands for recurrent neural network. It is a class of artificial neural network where connections between nodes form a directed graph along a sequence. 19

SemEval SemEval has evolved from the SensEval word sense disambiguation evaluation series. Every year it organizes challenges for several NLP task.. 11, 43, 48, 49, 58, 64, 94, 102, 103, 105, 106, 113

softmax Softmax is a generalized form of logistic regression for multi-class classification. It is very often used in output layer of a neural network models. 22, 80, 84, 85

SVM SVM stands for support vector machine. It is a linear classifier.. 2, 8, 34, 69, 78

TL Transfer learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. 4, 12, 95, 96, 98, 106, 114

UMLS The Unified Medical Language System (UMLS) is a compendium of many controlled vocabularies in the biomedical sciences (created 1986). It provides a mapping structure between the biomedical entities and it id.. 9

word2vec word2vec is a famous word embedding technique. It uses a large corpus to learn embeddings.. xi, 15, 27, 31

References

[Agarwal and Searls, 2008] Agarwal, P. and Searls, D. B. (2008). Literature mining in support of drug discovery. *Briefings in bioinformatics*, 9(6):479–492.

[Agichtein et al., 2005] Agichtein, E., Cucerzan, S., and Brill, E. (2005). Analysis of factoid questions for effective relation extraction. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 567–568. ACM.

[Ananiadou et al., 2006] Ananiadou, S., Kell, D. B., and ichi Tsujii, J. (2006). Text mining and its potential applications in systems biology. *Trends in Biotechnology*, 24(12):571 – 579.

[Aronson, 2001] Aronson, A. R. (2001). Effective mapping of biomedical text to the UMLS metathesaurus: the MetaMap program. In *Proceedings of the AMIA Symposium*, page 17. American Medical Informatics Association.

[Azizpour et al., 2015] Azizpour, H., Sharif Razavian, A., Sullivan, J., Maki, A., and Carlsson, S. (2015). From generic to specific deep representations for visual recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 36–45.

[Bahdanau et al., 2014] Bahdanau, D., Cho, K., and Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473.

[Bengio et al., 2013] Bengio, Y., Boulanger-Lewandowski, N., and Pascanu, R. (2013). Advances in optimizing recurrent networks. In *Acoustics, Speech and Signal Processing (ICASSP)*, pages 8624–8628. IEEE.

[Bengio et al., 2003] Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

[Bengio et al., 2007] Bengio, Y., Lamblin, P., Popovici, D., Larochelle, H., et al. (2007). Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153.

[Bengio et al., 1994] Bengio, Y., Simard, P., and Frasconi, P. (1994). Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

[Björne et al., 2013] Björne, J., Kaewphan, S., and Salakoski, T. (2013). UTurku: Drug named entity recognition and Drug-Drug interaction extraction using SVM classification and domain knowledge. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 651–659, Atlanta, Georgia, USA. Association for Computational Linguistics.

[Bobic et al., 2013] Bobic, T., Fluck, J., and Hofmann-Apitius, M. (2013). SCAI: Extracting drug-drug interactions using a rich feature vector. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 675–683, Atlanta, Georgia, USA. Association for Computational Linguistics.

[Bokharaeian and DIAZ, 2013] Bokharaeian, B. and DIAZ, A. (2013). NIL_UCM: Extracting Drug-Drug interactions from text through combination of sequence

and tree kernels. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 644–650, Atlanta, Georgia, USA. Association for Computational Linguistics.

[Bravo et al., 2015] Bravo, À., Piñero, J., Queralt-Rosinach, N., Rautschka, M., and Furlong, L. I. (2015). Extraction of relations between genes and diseases from text and large-scale data analysis: implications for translational research. *BMC bioinformatics*, 16(1):1.

[Bundschus et al., 2008] Bundschus, M., Dejori, M., Stetter, M., Tresp, V., and Kriegel, H.-P. (2008). Extraction of semantic biomedical relations from text using conditional random fields. *BMC bioinformatics*, 9(1):1.

[Bunescu et al., 2006] Bunescu, R., Mooney, R., Ramani, A., and Marcotte, E. (2006). Integrating co-occurrence statistics with information extraction for robust retrieval of protein interactions from medline. In *Proceedings of the Workshop on Linking Natural Language Processing and Biology: Towards Deeper Biological Literature Analysis*, pages 49–56. Association for Computational Linguistics.

[Businaro, 2013] Businaro, R. (2013). Why we need an efficient and careful pharmacovigilance. *J Pharmacovigilance*, 1(4):1000e110.

[Chiu and Nichols, 2015] Chiu, J. P. C. and Nichols, E. (2015). Named entity recognition with bidirectional LSTM-CNNs. *CoRR*, abs/1511.08308.

[Chowdhury and Lavelli, 2013a] Chowdhury, M. F. M. and Lavelli, A. (2013a). Exploiting the scope of negations and heterogeneous features for relation extraction: A case study for Drug-Drug interaction extraction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational*

Linguistics: Human Language Technologies, pages 765–771, Atlanta, Georgia. Association for Computational Linguistics.

[Chowdhury and Lavelli, 2013b] Chowdhury, M. F. M. and Lavelli, A. (2013b). FBK-irst : A Multi-Phase kernel based approach for Drug-Drug interaction detection and classification that exploits linguistic information. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 351–355, Atlanta, Georgia, USA. Association for Computational Linguistics.

[Collobert and Weston, 2008] Collobert, R. and Weston, J. (2008). A unified architecture for natural language processing: Deep neural networks with multitask learning. In *International Conference on Machine Learning*, pages 160–167. ACM.

[Collobert et al., 2011] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3):273–297.

[Cristianini and Shawe-Taylor, 2000] Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods*. Cambridge University Press, New York, NY, USA.

[Culotta and Sorensen, 2004] Culotta, A. and Sorensen, J. (2004). Dependency tree kernels for relation extraction. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 423. Association for Computational Linguistics.

[Dahl et al., 2012] Dahl, G. E., Yu, D., Deng, L., and Acero, A. (2012). Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. *IEEE Transactions on Audio, Speech, and Language Processing*, 20(1):30–42.

[Dogan et al., 2014] Dogan, R. I., Leaman, R., and Lu, Z. (2014). NCBI disease corpus: A resource for disease name recognition and concept normalization. *Journal of Biomedical Informatics*, 47:1–10.

[dos Santos and Guimarães, 2015] dos Santos, C. and Guimarães, V. (2015). Boosting named entity recognition with neural character embeddings. In *Proceedings of the Fifth Named Entity Workshop*, pages 25–33, Beijing, China. Association for Computational Linguistics.

[dos Santos and Zadrozny, 2014] dos Santos, C. N. and Zadrozny, B. (2014). Learning character-level representations for Part-of-Speech tagging. In *International Conference on Machine Learning (ICML)*, volume 32, pages 1818–1826. JMLR W&CP.

[Doğan and Lu, 2012] Doğan, R. I. and Lu, Z. (2012). An improved corpus of disease mentions in PubMed citations. In *Proceedings of the 2012 Workshop on Biomedical Natural Language Processing*, BioNLP '12, pages 91–99, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Duchi et al., 2011] Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *J. Mach. Learn. Res.*, 12(Jul):2121–2159.

[Farmakiotou et al., 2000] Farmakiotou, D., Karkaletsis, V., Koutsias, J., Sigletos, G., Spyropoulos, C. D., and Stamatopoulos, P. (2000). Rule-based named entity recognition for greek financial texts. In *Proceedings of the Workshop on*

Computational lexicography and Multimedia Dictionaries (COMLEX 2000), pages 75–78.

[Fleischman et al., 2003] Fleischman, M., Hovy, E., and Echihabi, A. (2003). Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 1–7, Sapporo, Japan. Association for Computational Linguistics.

[Girju, 2003] Girju, R. (2003). Automatic detection of causal relations for question answering. In *Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering-Volume 12*, pages 76–83. Association for Computational Linguistics.

[Graves, 2013] Graves, A. (2013). *Generating sequences with recurrent neural networks*. PhD thesis.

[Grego et al., 2013] Grego, T., Pinto, F., and Couto, F. M. (2013). LASIGE: using conditional random fields and chebi ontology. In *Proceedings of the 7th International Workshop on Semantic Evaluation*, pages 660–666.

[Gurulingappa et al., 2012a] Gurulingappa, H., Mateen-Rajpu, A., and Toldo, L. (2012a). Extraction of potential adverse drug events from medical case reports. *Journal of biomedical semantics*, 3(1):15.

[Gurulingappa et al., 2012b] Gurulingappa, H., Rajput, A. M., Roberts, A., Fluck, J., Hofmann-Apitius, M., and Toldo, L. (2012b). Development of a benchmark corpus to support the automatic extraction of drug-related adverse effects from medical case reports. *Journal of biomedical informatics*, 45(5):885–892.

[Harpaz et al., 2014] Harpaz, R., Callahan, A., Tamang, S., Low, Y., Odgers, D., Finlayson, S., Jung, K., LePendu, P., and Shah, N. H. (2014). Text mining for

adverse drug events: the promise, challenges, and state of the art. *Drug safety*, 37(10):777–790.

[Herrero-Zazo et al., 2013] Herrero-Zazo, M., Segura-Bedmar, I., Martnez, P., and Declerck, T. (2013). The DDI corpus: An annotated corpus with pharmacological substances and drugdrug interactions. *Journal of Biomedical Informatics*, 46(5):914 – 920.

[Hinton et al., 2012] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97.

[Hinton et al., 2006] Hinton, G. E., Osindero, S., and Teh, Y.-W. (2006). A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554.

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long Short-Term memory. *Neural Comput.*, 9(8):1735–1780.

[Hoeffding, 1952] Hoeffding, W. (1952). The large-sample power of tests based on permutations of observations. *The Annals of Mathematical Statistics*, pages 169–192.

[Hong, 2005] Hong, G. (2005). Relation extraction using support vector machine. In *Natural Language Processing–IJCNLP 2005*, pages 366–377. Springer.

[Hu et al., 2014] Hu, B., Lu, Z., Li, H., and Chen, Q. (2014). Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*, pages 2042–2050.

[Jiang et al., 2011] Jiang, M., Chen, Y., Liu, M., Rosenbloom, S. T., Mani, S., Denny, J. C., and Xu, H. (2011). A study of machine-learning-based approaches

to extract clinical entities and their assertions from discharge summaries. *Journal of the American Medical Informatics Association*, 18(5):601–606.

[Jurafsky, 2000] Jurafsky, D. (2000). *Speech & language processing*. Pearson Education India.

[Kalchbrenner et al., 2014] Kalchbrenner, N., Grefenstette, E., and Blunsom, P. (2014). A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland. Association for Computational Linguistics.

[Karpathy and Fei-Fei, 2014] Karpathy, A. and Fei-Fei, L. (2014). Deep visual-semantic alignments for generating image descriptions. *arXiv preprint arXiv:1412.2306*.

[Karpathy et al., 2015] Karpathy, A., Johnson, J., and Li, F. (2015). Visualizing and understanding recurrent networks. *CoRR*, abs/1506.02078.

[Kim et al., 2015] Kim, S., Liu, H., Yeganova, L., and Wilbur, W. J. (2015). Extracting drug–drug interactions from literature using a rich feature-based linear kernel approach. *Journal of biomedical informatics*, 55:23–30.

[Kim, 2014] Kim, Y. (2014). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, Doha, Qatar. Association for Computational Linguistics.

[Kingma and Ba, 2014] Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Ko, 2012] Ko, Y. (2012). A study of term weighting schemes using class information for text classification. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 1029–1030. ACM.

[Koehn, 2004] Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In *Proceedings of the 2004 conference on empirical methods in natural language processing*.

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, pages 1097–1105.

[Lafferty et al., 2001] Lafferty, J. D., McCallum, A., and Pereira, F. C. N. (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *International Conference on Machine Learning*, pages 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

[Lample et al., 2016] Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

[Leaman and Gonzalez, 2008] Leaman, R. and Gonzalez, G. (2008). BANNER: An executable survey of advances in biomedical named entity recognition. In Altman, R. B., Dunker, A. K., Hunter, L., Murray, T., and Klein, T. E., editors, *Pacific Symposium on Biocomputing*, pages 652–663. World Scientific.

[Leaman et al., 2009] Leaman, R., Miller, C., and Gonzalez, G. (2009). Enabling recognition of diseases in biomedical text with machine learning: corpus and benchmark. *Symposium on Languages in Biology and Medicine*, 82(9).

[LeCun et al., 1998] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

[Lee et al., 2007] Lee, C., Hwang, Y.-G., and Jang, M.-G. (2007). Fine-grained named entity recognition and relation extraction for question answering. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 799–800. ACM.

[Leroy et al., 2003] Leroy, G., Chen, H., and Martinez, J. D. (2003). A shallow parser based on closed-class words to capture relations in biomedical text. *Journal of biomedical Informatics*, 36(3):145–158.

[Li, 2012] Li, G. (2012). Disease mention recognition using Soft-Margin SVM. *Training*, 593:5–148.

[Li et al., 2016] Li, J., Monroe, W., and Jurafsky, D. (2016). Understanding neural networks through representation erasure. *CoRR*, abs/1612.08220.

[Lim et al., 2018] Lim, S., Lee, K., and Kang, J. (2018). Drug drug interaction extraction from the literature using a recursive neural network. *PLOS ONE*, 13(1):1–17.

[Lin et al., 2003] Lin, D., Zhao, S., Qin, L., and Zhou, M. (2003). Identifying synonyms among distributionally similar words. In *IJCAI*, volume 3, pages 1492–1493.

[Liu et al., 2016a] Liu, M., Shi, J., Li, Z., Li, C., Zhu, J., and Liu, S. (2016a). Towards better analysis of deep convolutional neural networks. *CoRR*, abs/1604.07043.

[Liu et al., 2016b] Liu, S., Tang, B., Chen, Q., and Wang, X. (2016b). Drug-Drug interaction extraction via convolutional neural networks. *Computational and mathematical methods in medicine*, 2016:1–8.

[Mahbub Chowdhury and Lavelli, 2010] Mahbub Chowdhury, M. F. and Lavelli, A. (2010). Disease mention recognition with specific features. In *Proceedings of the 2010 Workshop on Biomedical Natural Language Processing*, BioNLP '10, pages 83–90, Stroudsburg, PA, USA. Association for Computational Linguistics.

[McCallum et al., 2000] McCallum, A., Freitag, D., and Pereira, F. C. (2000). Maximum entropy markov models for information extraction and segmentation. In *International Conference on Machine Learning*, volume 17, pages 591–598.

[Mikolov et al., 2013a] Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

[Mikolov et al., 2010] Mikolov, T., Karafiát, M., Burget, L., Cernocký, J., and Khudanpur, S. (2010). Recurrent neural network based language model. In *INTERSPEECH 2010, 11th Annual Conference of the International Speech Communication Association, Makuhari, Chiba, Japan, September 26-30, 2010*, pages 1045–1048.

[Mikolov et al., 2013b] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani,

Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

[Milanfar, 2013] Milanfar, P. (2013). A tour of modern image filtering: New insights and methods, both practical and theoretical. *IEEE Signal Processing Magazine*, 30(1):106–128.

[Minard et al., 2011] Minard, A.-L., Ligozat, A.-L., and Grau, B. (2011). Multi-class SVM for relation extraction from clinical reports. In *RANLP*, pages 604–609.

[Mintz et al., 2009] Mintz, M., Bills, S., Snow, R., and Jurafsky, D. (2009). Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 1003–1011, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Morin and Bengio, 2005] Morin, F. and Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In Cowell, R. G. and Ghahramani, Z., editors, *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*, pages 246–252. Society for Artificial Intelligence and Statistics.

[Mou et al., 2016] Mou, L., Meng, Z., Yan, R., Li, G., Xu, Y., Zhang, L., and Jin, Z. (2016). How transferable are neural networks in nlp applications? In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 479–489, Austin, Texas. Association for Computational Linguistics.

[Nguyen and Grishman, 2015] Nguyen, T. H. and Grishman, R. (2015). Relation extraction: Perspective from convolutional neural networks. In *Proceedings of the*

1st Workshop on Vector Space Modeling for Natural Language Processing, pages 39–48, Denver, Colorado. Association for Computational Linguistics.

[Pan and Yang, 2010] Pan, S. J. and Yang, Q. (2010). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359.

[Park et al., 2001] Park, J. C., Kim, H. S., and Kim, J.-J. (2001). Bidirectional incremental parsing for automatic pathway identification with combinatory categorial grammar. In *Pacific Symposium on Biocomputing*, volume 6, pages 396–407.

[Pascanu et al., 2012] Pascanu, R., Mikolov, T., and Bengio, Y. (2012). Understanding the exploding gradient problem. *CoRR*, abs/1211.5063.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

[Pyysalo et al., 2012] Pyysalo, S., Ohta, T., Miwa, M., Cho, H.-C., Tsujii, J., and Ananiadou, S. (2012). Event extraction across multiple levels of biological organization. *Bioinformatics*, 28(18):i575–i581.

[Qian and Zhou, 2012] Qian, L. and Zhou, G. (2012). Tree kernel-based protein-protein interaction extraction from biomedical literature. *Journal of Biomedical Informatics*, 45(3):535 – 543.

[Rabiner, 1989] Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286.

[Raj et al., 2017] Raj, D., SAHU, S., and Anand, A. (2017). Learning local and global contexts using a convolutional recurrent network model for relation classification in biomedical text. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 311–321, Vancouver, Canada. Association for Computational Linguistics.

[Rastegar-Mojarad et al., 2013] Rastegar-Mojarad, M., Boyce, R. D., and Prasad, R. (2013). UWM-TRIADS: Classifying Drug-Drug interactions with Two-Stage SVM and Post-Processing. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 667–674, Atlanta, Georgia, USA. Association for Computational Linguistics.

[Riedel et al., 2010] Riedel, S., Yao, L., and McCallum, A. (2010). Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163. Springer.

[Rink et al., 2011] Rink, B., Harabagiu, S., and Roberts, K. (2011). Automatic extraction of relations between medical concepts in clinical texts. *Journal of the American Medical Informatics Association*, 18(5):594–600.

[Rocktäschel et al., 2013] Rocktäschel, T., Huber, T., Weidlich, M., and Leser, U. (2013). WBI-NER: The impact of domain-specific features on the performance of identifying and classifying mentions of drugs. In *Second Joint Conference on Lexical and Computational Semantics (* SEM)*, volume 2, pages 356–363.

[Rocktäschel et al., 2012] Rocktäschel, T., Weidlich, M., and Leser, U. (2012). ChemSpot: a hybrid system for chemical named entity recognition. *Bioinformatics*, 28(12):1633–1640.

[Roller and Stevenson, 2014] Roller, R. and Stevenson, M. (2014). Applying UMLS for distantly supervised relation detection. In *Proceedings of the 5th International Workshop on Health Text Mining and Information Analysis (Louhi)*, pages 80–84, Gothenburg, Sweden. Association for Computational Linguistics.

[Rosario and Hearst, 2004] Rosario, B. and Hearst, M. A. (2004). Classifying semantic relations in bioscience texts. In *Proceedings of the 42Nd Annual Meeting on Association for Computational Linguistics*, ACL '04, Stroudsburg, PA, USA. Association for Computational Linguistics.

[Rosenblatt, 1958] Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386.

[Rumelhart et al., 1988] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1988). Neurocomputing: Foundations of research. chapter Learning Representations by Back-propagating Errors, pages 696–699. MIT Press, Cambridge, MA, USA.

[Sahu and Anand, 2016] Sahu, S. and Anand, A. (2016). Recurrent neural network models for disease name recognition using domain invariant features. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2216–2225, Berlin, Germany. Association for Computational Linguistics.

[Sahu et al., 2016] Sahu, S., Anand, A., Oruganty, K., and Gattu, M. (2016). Relation extraction from clinical texts using domain invariant convolutional neural network. In *Proceedings of the 15th Workshop on Biomedical Natural Language Processing*, pages 206–215, Berlin, Germany. Association for Computational Linguistics.

[Sahu and Anand, 2017a] Sahu, S. K. and Anand, A. (2017a). Drug-Drug Interaction Extraction from Biomedical Text Using Long Short Term Memory Network. *CoRR*, abs/1701.08303.

[Sahu and Anand, 2017b] Sahu, S. K. and Anand, A. (2017b). Unified Neural Architecture for Drug, Disease and Clinical Entity Recognition. *CoRR*, abs/1708.03447.

[Sahu and Anand, 2018] Sahu, S. K. and Anand, A. (2018). What matters in a transferable neural network model for relation classification in the biomedical domain? *Artificial Intelligence in Medicine*, 87:60 – 66.

[Segura-Bedmar et al., 2013] Segura-Bedmar, I., Martínez, P., and Herrero Zazo, M. (2013). SemEval-2013 Task 9 : Extraction of Drug-Drug Interactions from Biomedical Texts (DDIExtraction 2013). In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 341–350, Atlanta, Georgia, USA. Association for Computational Linguistics.

[Segura Bedmar et al., 2011] Segura Bedmar, I., Martinez, P., and Sánchez Cisneros, D. (2011). The 1st DDIExtraction-2011 challenge task: Extraction of Drug-Drug interactions from biomedical texts.

[Segura-Bedmar et al., 2015] Segura-Bedmar, I., Suárez-Paniagua, V., and Martínez, P. (2015). Exploring word embedding for drug name recognition. In *Proceedings of the Sixth International Workshop on Health Text Mining and Information Analysis*, pages 64–72, Lisbon, Portugal. Association for Computational Linguistics.

[Settles, 2004] Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. In Collier, N., Ruch, P., and Nazarenko, A., editors, *COLING 2004 International Joint workshop on Natural*

Language Processing in Biomedicine and its Applications (NLPBA/BioNLP) 2004, pages 107–110, Geneva, Switzerland. COLING.

[Settles, 2005] Settles, B. (2005). ABNER: An open source tool for automatically tagging genes, proteins, and other entity names in text. *Bioinformatics*, 21(14):3191–3192.

[Shang et al., 2011] Shang, Y., Li, Y., Lin, H., and Yang, Z. (2011). Enhancing Biomedical Text Summarization Using Semantic Relation Extraction. *PLoS ONE*, 6:1–10.

[Sharma et al., 2016] Sharma, R. D., Tripathi, S., Sahu, S. K., Mittal, S., and Anand, A. (2016). Predicting online doctor ratings from user reviews using convolutional neural networks. *International Journal of Machine Learning and Computing*, 6(2):149.

[Socher et al., 2012] Socher, R., Huval, B., Manning, C. D., and Ng, A. Y. (2012). Semantic compositionality through recursive Matrix-Vector spaces. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 1201–1211, Jeju Island, Korea. Association for Computational Linguistics.

[Socher et al., 2011] Socher, R., Lin., C. C.-Y., Manning, C. D., and Ng, A. Y. (2011). Parsing Natural Scenes and Natural Language with Recursive Neural Networks. In *ICML*.

[Song et al., 2011] Song, Q., Watanabe, Y., and Yokota, H. (2011). Relationship extraction methods based on co-occurrence in web pages and files. In *Proceedings of the 13th International Conference on Information Integration and Web-based Applications and Services*, iiWAS '11, pages 82–89, New York, NY, USA. ACM.

[Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

[Stapley and Benoit, 2000] Stapley, B. J. and Benoit, G. (2000). Biobibliometrics: information retrieval and visualization from co-occurrences of gene names in medline abstracts. In *Pac Symp Biocomput*, volume 5, pages 529–540.

[Suárez-Paniagua and Segura-Bedmar, 2016] Suárez-Paniagua, V. and Segura-Bedmar, I. (2016). Extraction of Drug-Drug interactions by recursive Matrix-Vector spaces. In *6thInternational Workshop on Combinations of Intelligent Methods and Applications (CIMA 2016)*, page 65.

[Surdeanu et al., 2012] Surdeanu, M., Tibshirani, J., Nallapati, R., and Manning, C. D. (2012). Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 455–465, Jeju Island, Korea. Association for Computational Linguistics.

[TH et al., 2015] TH, M., Sahu, S., and Anand, A. (2015). Evaluating distributed word representations for capturing semantics of biomedical concepts. In *Proceedings of the 14th Workshop on Biomedical Natural Language Processing*, pages 158–163, Beijing, China. Association for Computational Linguistics.

[Thomas et al., 2000] Thomas, J., Milward, D., Ouzounis, C., Pulman, S., and Carroll, M. (2000). Automatic extraction of protein interactions from scientific. In *Pacific symposium on biocomputing*, volume 5, pages 538–549.

[Thomas et al., 2013] Thomas, P., Neves, M., Rocktäschel, T., and Leser, U. (2013). WBI-DDI: Drug-Drug interaction extraction using majority voting. In *Second Joint Conference on Lexical and Computational Semantics (*SEM),*

Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013), pages 628–635, Atlanta, Georgia, USA. Association for Computational Linguistics.

[Tjong Kim Sang and De Meulder, 2003a] Tjong Kim Sang, E. F. and De Meulder, F. (2003a). Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics.

[Tjong Kim Sang and De Meulder, 2003b] Tjong Kim Sang, E. F. and De Meulder, F. (2003b). Introduction to the CoNLL-2003 Shared Task: Language-Independent Named Entity Recognition. In Daelemans, W. and Osborne, M., editors, *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147.

[Turian et al., 2010] Turian, J., Ratinov, L.-A., and Bengio, Y. (2010). Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden. Association for Computational Linguistics.

[Turney and Pantel, 2010] Turney, P. D. and Pantel, P. (2010). From frequency to meaning: Vector space models of semantics. *J. Artif. Int. Res.*, 37(1):141–188.

[Uzuner et al., 2011] Uzuner, Ö., South, B. R., Shen, S., and DuVall, S. L. (2011). 2010 i2b2/VA challenge on concepts, assertions, and relations in clinical text. *Journal of the American Medical Informatics Association*, 18(5):552–556.

[Uzuner et al., 2014] Uzuner, Ö., Yetisgen, M., and Stubbs, A. (2014). Biomedical/Clinical NLP. *COLING (Tutorials)*, pages 1–2.

[van Mulligen et al., 2012] van Mulligen, E. M., Fourrier-Reglat, A., Gurwitz, D., Molokhia, M., Nieto, A., Trifiro, G., Kors, J., and Furlong, L. I. (2012). The EU-ADR corpus: Annotated drugs, diseases, targets, and their relationships. *Journal of biomedical informatics*, 45(5):879–884.

[Wang et al., 2017] Wang, W., Yang, X., Yang, C., Guo, X., Zhang, X., and Wu, C. (2017). Dependency-based long short term memory network for drug-drug interaction extraction. *BMC bioinformatics*, 18(16):578.

[Xu, 2008] Xu, F.-Y. (2008). *Bootstrapping Relation Extraction from Semantic Seeds*. PhD thesis, Saarland University.

[Yang et al., 2017] Yang, Z., Salakhutdinov, R., and Cohen, W. W. (2017). Transfer learning for sequence tagging with hierarchical recurrent networks. *CoRR*, abs/1703.06345.

[Yao et al., 2015] Yao, L., Liu1, H., Liu, Y., Li, X., and Anwars, M. W. (2015). Biomedical named entity recognition based on deep neutral network. *International Journal of Hybrid Information Technology*, pages 279–288.

[Yosinski et al., 2014] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328.

[Yosinski et al., 2015] Yosinski, J., Clune, J., Nguyen, A. M., Fuchs, T. J., and Lipson, H. (2015). Understanding neural networks through deep visualization. *CoRR*, abs/1506.06579.

[Zelenko et al., 2003] Zelenko, D., Aone, C., and Richardella, A. (2003). Kernel methods for relation extraction. *J. Mach. Learn. Res.*, 3:1083–1106.

[Zeng et al., 2014] Zeng, D., Liu, K., Lai, S., Zhou, G., and Zhao, J. (2014). Relation classification via convolutional deep neural network. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 2335–2344.

[Zhang and Wang, 2015] Zhang, D. and Wang, D. (2015). Relation classification via recurrent neural network. *arXiv preprint arXiv:1508.01006*.

[Zhang et al., 2015] Zhang, X., Zhao, J., and LeCun, Y. (2015). Character-level convolutional networks for text classification. In *Advances in neural information processing systems*, pages 649–657.

[Zhao et al., 2016] Zhao, Z., Yang, Z., Luo, L., Lin, H., and Wang, J. (2016). Drug drug interaction extraction from biomedical literature using syntax convolutional neural network. *Bioinformatics*, 32(22):3444–3453.

[Zhou et al., 2016] Zhou, P., Shi, W., Tian, J., Qi, Z., Li, B., Hao, H., and Xu, B. (2016). Attention-Based bidirectional long Short-Term memory networks for relation classification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 207–212, Berlin, Germany. Association for Computational Linguistics.

[Zoph et al., 2016] Zoph, B., Yuret, D., May, J., and Knight, K. (2016). Transfer learning for Low-Resource neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1568–1575, Austin, Texas. Association for Computational Linguistics.

Brief Biography of the Author

Sunil Kumar Sahu was born in Korba, Chhattisgarh, India on 26 September 1987. After completing his schooling in Higher Secondary School Ghordeva, he received the Bachelor of Engineering (BE) degree from *Department of Computer Science and Engineering, Gurughasidas University, Bilaspur (CG)*, India in 2009. He completed his M.Tech. degree from *School of Computer and Information Science, University of Hyderabad, Hyderabad* in 2013, and after that joined *Department of Computer Science and Engineering, Indian Institute of Technology (IIT) Guwahati* as a Ph.D. research scholar. During his Ph.D., he has been supervised by Dr. Ashish Anand. His research interests include Natural Language Processing, Machine Learning, Neural Network and Data Mining.

Publications Related to Thesis

Journals

1. Sunil Kumar Sahu, Ashish Anand, *"What matters in a transferable neural network model for relation classification in the biomedical domain?"*, Artificial Intelligence in Medicine, Elsevier

2. Sunil Kumar Sahu, Ashish Anand, *"Unified Neural Architecture for Drug, Disease and Clinical Entity Recognition"*, Artificial Intelligence in Medicine, Elsevier (**Major revision submitted**)

3. Sunil Kumar Sahu, Ashish Anand, *"Drug-Drug Interaction Extraction from Biomedical Text Using Long Short Term Memory Network"*, Journal of Biomedical and Informatics, Elsevier (**Major revision submitted**)

Conferences

1. Desh Raj, **Sunil Kumar Sahu**, Ashish Anand, *"Learning local and global contexts using a convolutional recurrent network model for relation classification in biomedical text"*, CoNLL-2017, Vancouver, Canada, Aug-2017

2. Sunil Kumar Sahu, Ashish Anand, Krishnadev Oruganty, Mahanandeeshwar Gattu, *"Relation extraction from clinical texts using domain invariant convolutional neural network"*, BioNLP at ACL-2016, Berlin, Germany, Aug-2016

3. Sunil Kumar Sahu, Ashish Anand, *"Recurrent neural network models for disease name recognition using domain invariant features"*, ACL-2016, Berlin, Germany, Aug-2016

4. Muneeb TH, **Sunil Kumar Sahu**, Ashish Anand, *"Evaluating distributed word representations for capturing semantics of biomedical concepts"*, BioNLP at ACL-2015, Beijing, China, July-Aug 2015

Publications Outside Thesis

1. Patchigolla V S S Rahul, **Sunil Kumar Sahu**, Ashish Anand, *"Biomedical event trigger identification using bidirectional recurrent neural network based models"*, BioNLP at ACL-2017, Vancouver, Canada, Aug-2017

2. Ranti D Sharma, Samarth Tripathi, **Sunil Kumar Sahu**, Sudhanshu Mittal, Ashish Anand, *"Predicting online doctor ratings from user reviews using convolutional neural networks"*, International Journal of Machine Learning and Computing (IJMLC), Vol. 6, No. 2, April 2016

Department of Computer Science and Engineering

Indian Institute of Technology Guwahati

Guwahati 781039, India